Improving Real World Performance for Vision Navigation in a Flight Environment

DISSERTATION

Donald Thomas Venable, Civilian

AFIT-ENG-DS-16-S-017

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY

AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

AFIT-ENG-DS-16-S-017

Improving Real World Performance for Vision Navigation in a Flight Environment

DISSERTATION

Presented to the Faculty

Graduate School of Engineering and Management

Air Force Institute of Technology

Air University

Air Education and Training Command

In Partial Fulfillment of the Requirements for the

Degree of Doctor of Philosophy

Donald Thomas Venable, B.S.E.E., M.S.E.E

Civilian

September, 2016

AFIT-ENG-DS-16-S-017

Improving Real World Performance for Vision Navigation in a Flight Environment

Donald Thomas Venable, B.S.E.E., M.S.E.E

Civilian

AFIT-ENG-DS-16-S-017

Abstract

The advent of the Global Positioning System has redefined the role of navigation for both commercial and military consumers. Current civil and military applications depend heavily on the Global Positioning System, and the realization of this dependence has caused application designers to seek alternatives and backups for situations where GPS is unavailable. One category of alternatives seeks to find cost-effective alternatives for military systems by leveraging information from non-navigation sensors that are already deployed on fielded platforms. The motivation of this research is to fuse information from an airborne imaging sensor with information extracted from satellite imagery in order to provide accurate global position in scenarios where GPS is unavailable for an extend duration. The research outlines the architecture for an Image Positioning System, composing of three elements: a position acquisition algorithm, a fine-tracking algorithm, and a database of image keypoints that supports both estimation components. This research provides a description of the image processing pipeline, where multiple types of point features are extracted from imagery. These keypoints are computed from a corpus of existing georeferenced satellite and airborne imagery to create a keypoint database. An offline algorithm utilizes the database in order to predict which keypoints will be most useful during the online navigation algorithms. The first step in the online portion of the airborne image positioning system is to recover coarse position when faced with a large initial position uncertainty. This research details a novel algorithm for recovering coarse pose using by comparing keypoints extracted from the airborne imagery to the reference database. This coarse position is used to bootstrap a local-area georegistration algorithm, which provides GPS-level position estimates. This research derives optimizations for existing local-area methods for operation in flight environments. This research evaluates the components of the Image Positioning System using data from flight experiments.

For Margaret

Table of Contents

List of Figures

Figure Page

Figure Page

Figure Page

List of Tables

List of Abbreviations

Abbreviation		Page
PNT	Positioning, Navigation, and Timing	1
GPS	Global Positioning System	1
IMU	Inertial Measurement Unit	1
INS	Inertial Navigation System	1
EKF	Extended Kalman Filter	1
EO	Electro-Optical	2
IR	Infrared	2
SLAM	Simultaneous Localization and Mapping	2
IPS	Image Positioning System	4
DOF	Degrees of Freedom	5
ECI	Earth-Centered Inertial	8
ECEF	Earth-Centered Earth-Fixed	8
WGS-84	World Geodetic System -1984	9
DEM	Digital Elevation Model	9
MSL	Mean Sea Level	9
MMSE	Minimum Mean Squared Error	14
EKF	Extended Kalman filter	15
ADC	Analog to Digital Converter	19
FPA	Focal Plane Array	19
PSF	Point-Spread Function	19
GSD	Ground Sample Distance	26
IFOV	Instantaneous Field Of View	26
SIFT	Scale Invariant Feature Transform	28
FAST	Features from Accelerated Segment Test	28
HoG	Histogram of Oriented Gradients	28

Improving Real World Performance for Vision Navigation in a Flight Environment

I. Introduction

Positioning, Navigation and Timing (PNT) has become a critical tenet of the modern infrastructure, proliferating into our daily lives. PNT has had a significant impact on the world by enabling the core functionality of smartphone applications, emergency services, and localization of airborne platforms, mainly from widespread adoption of the Global Positioning System (GPS),

In the case of airborne navigation, commercial airliners and most defense platforms rely on the integration of a navigation solution from an Inertial Measurement Unit (IMU). The IMU provides measurements of specific force and angular velocity. These measurements are used in a strap-down Inertial Navigation System (INS) mechanization to provide a navigation solution [2]. However, error in the IMU accelerometers and gyroscopes, approximations in the strap-down INS equations, and error initial conditions lead to a position solution that grows unbounded over time [3]. IMUs are typically divided into 3 grades: navigation, tactical, and consumer [2]. Navigation grade IMUs, when calibrated, will typically result in a 1 nautical mile / hr drift. A Monte-Carlo simulation in [4] shows that 3D RSS position error of an INS using a tactical-grade IMU to be roughly 10 km over a period of 20 minutes. The research in this paper is focused on recovering from the position error incurred by use of a tactical-grade IMU over a 30 minute or longer period, with results in a horizontal search area of greater than 1000 km^2.

To bound the position solution, airborne platforms typically integrate measurements from a GPS receiver using a recursive Bayes estimator such as the Extended Kalman Filter (EKF). The Extended Kalman Filter is used to estimate the error parameters of the IMU, and these are fed back to to the INS. When GPS is then degraded or denied, the platform uses the most current estimate of the IMU errors. However, the position solution error will then start an unbounded drift until additional aiding measurements are available. Therefore "alternative" sensors may be used to compute information about the navigation state of the platform during GPS outages. Historically, sensors such as cameras [4], sonar [5], TV/Radio Signals [6], Cell phone towers [7], LiDAR [8], and RADAR [9] have been used to aid the navigation solution, each with their own benefits and weaknesses.

1.1 Vision Navigation in a Flight Environment

Electro-Optical (EO) and infrared (IR) cameras are sensors commonly employed for many types of applications including smartphones, automobiles, robotic platforms, and aircraft. Due to the availability of EO/IR cameras, and the quantity of data coming from these sensors, significant work has been undertaken to extract information about the navigation state from image data. Currently, many approaches employ some scheme of Simultaneous Localization and Mapping (SLAM) [4] [10] wherein estimates of a vehicle's pose are jointly estimated along with locations of observed salient "landmarks" derived from the imagery. The current state-of-the-art approaches utilize existing SLAM techniques for online localization with some auxiliary algorithms to detect instances when the platform is revisiting a previously explored area. These auxiliary algorithms are commonly referred to as place recognition. Many approaches draw from recent research in the Content Based Image Retrieval (CBIR) systems [11] [12] or schemes based on approximate nearest neighbor search of feature descriptors [13].

CBIR systems typically rely on creation of a vison-vocabulary, created by clustering landmarks with similar appearance metrics into visual words. New images are compared to old ones by ranking their similarity through heuristic metrics [11] or approximate Bayesian probability [12]. While very effective for performing place detection for a single ground platform, creating a clustering of landmarks useful for matching images across a wide area of geography or multiple sensors has been shown to be difficult [14]. Several approximations are made in [12] to incorporate longer mission durations which will not scale infinitely and may not be suitable for military applications. Additionally, CBIR systems currently operate only in appearance space and do not fully incorporate metric information available from other onboard sensors. The proposed research aims to leverage the progress made in SLAM and CBIR systems by developing a probabilistic CBIR system geared toward inference over sets of imagery collected by multiple platforms, separated spatially and temporally.

1.2 Problem Definition

The process of navigation is defined as determining a relative translation and orientation from a reference frame with respect to time [4]. Vision-aided navigation is defined as the use of measurements derived from EO imagery to infer navigation state. This dissertation defines georegistration-based image-aided navigation as the use of a reference set of orthorectified imagery to compute a global position estimate of an airborne platform. This research aims to address two deficiencies in the state of the art in current vision-aided navigation techniques: position acquisition and image landmark databases.

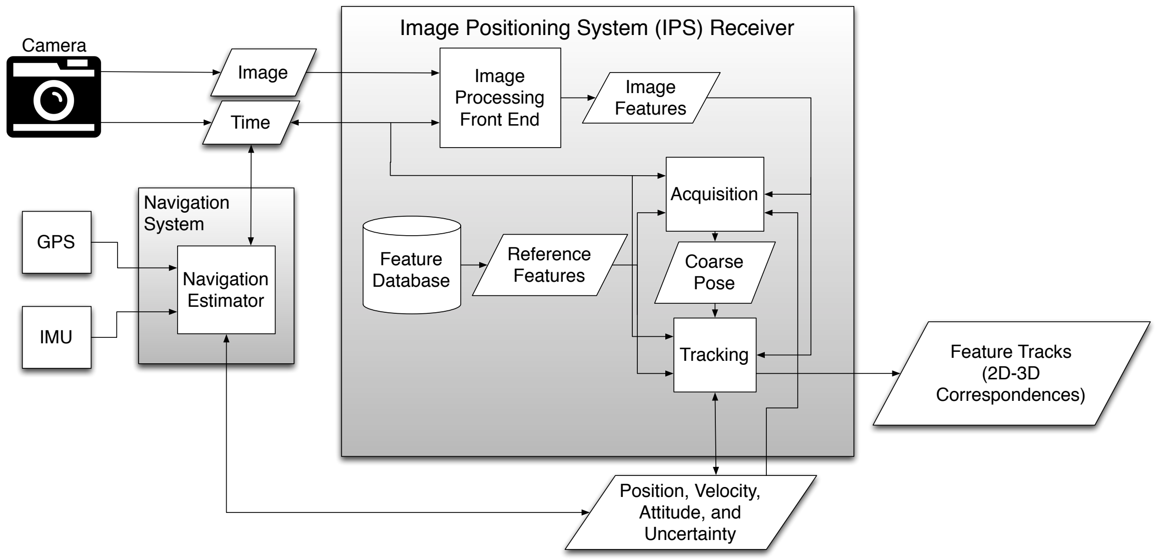
1. **Position Acquisition**: In situations where an airborne platform is denied access to navigation signals for an extended duration, the position uncertainty may be too large to re-acquire position using existing vision-based techniques. Image measurements suitable for state inference in vision-aided navigation may become unavailable due to cloud cover, platform maneuvering, or observation of non-distinct terrain (*e.g.*, water, desert). This research proposes an algorithm that uses platform imagery and a reference database to compute an estimate of aircraft position with an uncertainty suitable to bootstrap existing vision-aided navigation methods.
2. **Image Landmark Databases**: Georegistation-based vision-aided navigation techniques leverage some form of *a priori* database of georeferenced imagery, image keypoints and their descriptors, or vision-vocabulary statistics. However,

there is currently a gap in the literature for rigorously creating, maintaining, and utilizing landmark databases for vision-aided navigation.

1.3 Research Contributions

This research is focused on providing a system that is able to provide absolute position updates to an airborne platform by comparing information extracted from airborne imagery to a reference database. The algorithms developed in this research form the key components of the Image Positioning System (IPS) Receiver, a conceptual data-flow that estimates aircraft position given platform imagery and a reference database. A system diagram of the IPS is given in Figure 1. The research in this dissertation provides the following contributions in order to realize the implementation of the Image Positioning System:

- Development of a novel image keypoint database. This database is created from orthorectified reference imagery, and continuously refined using post-processed airborne imagery. The database also supports an offline machine-learning process to predict which keypoints are most useful for the online navigation algorithms detailed in this dissertation.
- The largest contribution is the development and implementation of a novel wide-area search algorithm. This algorithm is designed to provide an image acquisition function, where a rough estimate of aircraft position is provided, to recover from a large prior position uncertainty. This algorithm implements a histogram filter that uses a novel measurement likelihood function, which is approximated using statistics of matches between features derived from the airborne platform and the reference database. We subsample the keypoints loaded from the reference database using the metadata computed in the offline machine-learning step. The database is further subsampled using a keypoint scale filter.
- Implementation of an image-based georegistration function, which provides GPS-like position information to an external estimator. This function uses the prior



Camera
Image
Time
Image Positioning System (IPS) Receiver
Image Processing Front End
Image Features
Acquisition
Coarse Pose
Tracking
Feature Database
Reference Features
Navigation System
Navigation Estimator
GPS
IMU
Feature Tracks (2D-3D Correspondences)
Position, Velocity, Attitude, and Uncertainty

Figure 1: System Diagram of the Image Positioning System

position information from either the acquisition algorithm, or the legacy navigation system in order to load a subset of the keypoints for navigation. This research shows that careful sampling of keypoints using metadata provided by the database provides a significant matching benefit. Furthermore, it is shown that using the attitude estimate from the INS to constrain the 6 Degrees-of-Freedom DOF position and attitude estimation problem to 3-DOF provides significant position accuracy improvement.

1.4 Dissertation Outline

This dissertation is organized as follows. Chapter II proves a mathematical background for navigation state estimation, imaging scenarios, image keypoint extraction and description, and coordinate systems used in the research. Additionally, an analysis of the current state of the art in vision-aided navigation is provided. Chapter III outlines the image-keypoint database, detailing the creation of the database from historical sets of satellite and airborne imagery, and heuristics used to rank keypoints on their usefulness for navigation. The use of this database to provide a coarse position estimate to an airborne platform operating in a GPS-challenged environment for a significant duration is provided in Chapter IV. Chapter V outlines the improvements to the state of the art for local-area vision-aided navigation techniques for implementation in a flight environment. Finally, the conclusions of the research provided in this dissertation, along with suggestions for future work are given in Chapter VI.

II. Relevant Background and Literature Review

This chapter provides a mathematical background for the research presented in later chapters of the dissertation. First, an overview of the notation used in the dissertation is presented. The next section provides an overview of the common coordinate systems and reference frames encountered in this research. Next, the concept of recursive Bayesian estimation is presented to provide context for the wide-area search algorithm that is presented in Chapter IV. Finally, a detailed background on EO imaging, image scale decomposition, creation of image features, feature descriptor matching, and the feature descriptor clustering techniques used in content-based image retrieval is presented.

2.1 Mathematical Notation

The following conventions are used for the notation of mathematical elements in this research:

- **Scalars**: Scalars are represented by letters in italic type, (e.g., x or P).
- **Vectors**: Vectors are denoted by lower case letters in bold type, (e.g., $\mathbf{x}$ or $\boldsymbol{\phi}$). Scalar elements of the vector are referenced by their index element, (e.g., x_i is the i^{th} element of $\mathbf{x}$).
- **Matrices**: Matrices are represented by upper case letters in bold type, (e.g., $\mathbf{X}$). The elements of a matrix are denoted by upper case letters in italics, with subscripts for the row and column indices. For example, X_{ij} is the element from the i^{th} row and j^{th} column of $\mathbf{X}$.
- **Transpose**: The transpose of a vector or matrix is denoted with a superscript T, (i.e., $\mathbf{x}^T$ is the transpose of $\mathbf{x}$).
- **Estimated Variables**: Variables representing the estimated value of a random variable are annotated with a hat, as in $\hat{\mathbf{x}}$.
- **Rotation Matrices**: Matrices that represent a rotation from frame a to b are denoted as $\mathbf{R}_a^b$.

- **Reference Frame**: If a vector quantity is expressed with respect to a reference frame, the variable is annotated with a superscript. For example, $\mathbf{x}^b$ represents a vector quantity with respect to b.

2.2 Coordinate Systems

This thesis will be concerned with calculating the navigation state of an aerial vehicle. The main components of the navigation state are the translation vector and a rotation of the aircraft with respect to a reference coordinate system. The 3-dimensional translation of a rigid body is represented as a vector t^i, or the displacement from the origin of the Euclidean coordinate system i, expressed in coordinates of i. Rotation is defined as a 3×3 matrix from the special orthogonal group SO(3). A rotation of a platform frame j, about the coordinate system i, is denote as $\mathbf{R}^i_j$. The transformation of a translation vector represented in the coordinate system of j to a representation in i is given by:

$$\mathbf{t}^i = \mathbf{R}^i_j \mathbf{t}^j + \mathbf{j}^i_0 \tag{1}$$

where $\mathbf{j}^i_0$ is the translation of the origin of coordinate system j referenced in coordinate system i. Common Euclidean coordinate systems used in navigation such as Earth-Centered Inertial (ECI), Earth-Centered Earth-Fixed (ECEF), and various Local-Level frames are described in [2].

The aircraft body frame is an Euclidean coordinate system whose origin is at the center of gravity of the aircraft. The x^b, y^b, and z^b axes of the aircraft body frame point out the nose, right wing, and bottom of the aircraft, respectively. An illustration of the aircraft body frame is provided in Figure 2. The airborne camera also has a Euclidean coordinate system, given in Section 2.4.3. Normally, there is a rigid translation, t^b_{cam}, and rotation, $\mathbf{R}^b_{\text{cam}}$ between the body and camera coordinate systems, defined as the camera extrinsic calibration, or lever-arm.

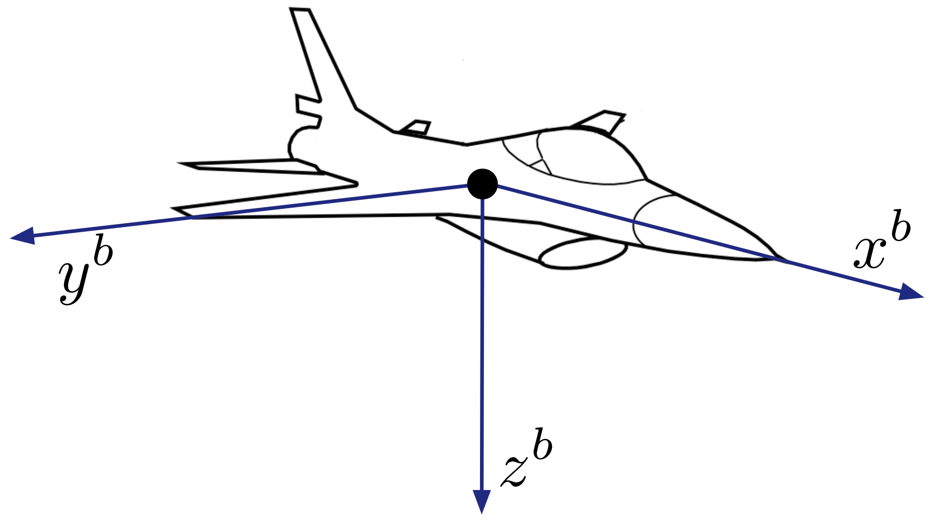


Figure 2: Illustration of aircraft body frame.

This research also uses an ellipsoidal coordinate system to represent global position. The Department of Defense has created a standardized ellipsoid model of the earth, called the World Geodetic System - 1984 (WGS-84) [15]. WGS-84 is the global reference frame used by GPS, and is the global reference frame used in this research. Conversion between WGS-84 and the various Euclidean coordinate systems use a non-linear transformation [2]. In addition, this research also utilizes terrain height information from a Digital Elevation Model (DEM). Terrain height typically uses Mean Sea Level (MSL) as a vertical reference. Mean Sea Level is modeled non-uniformly across the ellipsoid model of the world, and the vertical displacement between the ellipsoid and MSL at any given point on the ellipsoid is referred to as undulation. Various organizations provide world-wide models of undulation which can be used to convert height values represented in MSL to height above the WGS-84 ellipsoid [15].

This research also utilizes the Spherical Mercator coordinate system to create unique indices for keypoints extracted from georeferenced imagery. Spherical Mercator uses a cylindrical projection to create a 2D mapping of a spherical world model, where lines of latitude are mapped to horizontal lines, and lines of latitude are represented as parallel vertical lines. Spherical Mercator also defines a notion of scale/zoom, where the projected sphere is divided into a quadtree representation. This coordinate system is used extensively in Internet mapping systems [16]. While distortion error is introduced by both the spherical approximation of the WGS-84 ellipsoid, and the Mercator projection of that sphere, this coordinate system provides a simple method for defining a grid of varying scale across the globe. For a desired zoom level in the quadtree, z, the tile that contains a point on the globe with longitude λ and latitude ϕ in radians is given as [16]:

$$t_x = \left\lfloor \frac{\lambda + \pi}{2\pi} 2^z \right\rfloor \tag{2}$$

$$t_y = \left\lfloor \left(1 - \frac{\ln\left(\tan(\phi) + \frac{1}{\cos(\phi)}\right)}{\pi} \right) 2^{z-1} \right\rfloor \tag{3}$$

where t_x and t_y are the x and y coordinates of the tile in the Spherical Mercator coordinate system, respectively.

2.3 Estimation

This research is concerned with estimation of the navigation state vector. This vector consists of random variables containing information about the pose (position and attitude) of the aerial platform at time t_k, denoted $\mathbf{x}_{t_k}$. Estimates of this state vector, denoted $\hat{\mathbf{x}}_{t_k}$ will be evaluated using observations / measurements from the platform over time. A single observation at the discrete time t_k is denoted as $\mathbf{z}_{t_k}$. The collection of measurements from the initial time until time t_k is denoted as $\mathbf{Z}_{t_k}$. The system may be actively controlled over time, and these inputs are represented as $\mathbf{u}_{t_k}$. Analogous to measurements, the collection of these inputs is given as: $\mathbf{U}_{t_k} = \mathbf{u}_0 : \mathbf{u}_{t_k}$

When estimating the real-time navigation state of an aircraft, typical estimation schemes implement *recursive* state estimation, where the current state vector $\mathbf{x}_{t_k}$ is estimated using the prior state estimate $\mathbf{x}_{t_{k-1}}$, and incorporation of the most current measurement/observation, $\mathbf{z}_{t_k}$. Methods for accomplishing this estimation process are described next, where most of derivation and terminology is derived from [10], with some notation changes which may be more familiar to those associated with [17].

2.3.1 Recursive Bayesian Estimation. A popular formulation of recursive state estimation is known as the *Bayes filter* or *Bayesian Recursion.* The Bayes filter aims to calculate the belief about the vector of random variables that compose the state vector, which can be represented as a posterior distribution function $p(\mathbf{x}_{t_k})$. When calculated as being explicitly conditioned on all past measurements and control inputs, this belief function is represented as:

$$\text{bel}(\mathbf{x}_{t_k}^{+}) = p(\mathbf{x}_{t_k}^{+}|\mathbf{Z}_{t_k}, \mathbf{U}_{t_k}) \tag{4}$$

The prediction of the state estimate belief calculated at time t_k, directly before incorporating the measurement at time t_k, $\mathbf{z}_{t_k}$, is represented as:

$$\text{bel}(\mathbf{x}_{t_k}^{-}) = p(\mathbf{x}_{t_k}^{-}|\mathbf{Z}_{t_{k-1}}, \mathbf{U}_{t_k}) \tag{5}$$

A key assumption made is that of the navigation state being ‘complete’, also known as the Markov assumption, making the claim that the past and future observations are independent of each other conditioned on the current state $\mathbf{x}_{t_k}$.

Bayes Rule is used to calculate the posterior distribution using the probability distribution function for both the prediction step and the measurement observation, shown as:

$$p(\mathbf{x}_{t_k}^{+}|\mathbf{Z}_{t_k}, \mathbf{U}_{t_k}) = \frac{p(\mathbf{z}_{t_k}|\mathbf{x}_{t_k}, \mathbf{Z}_{t_{k-1}}, \mathbf{U}_{t_k})p(\mathbf{x}_{t_k}^{-}|\mathbf{Z}_{t_{k-1}}, \mathbf{U}_{t_k})}{p(\mathbf{z}_{t_k}|\mathbf{Z}_{t_{k-1}}, \mathbf{U}_{t_k})} \tag{6}$$

The subsequent sections will describe applications of recursive Bayes filtering for specific types of estimation problems.

2.3.1.1 Kalman filter. A very widely implementation of the recursive Bayes filter is the Kalman filter. The Kalman filter is a formulation of the Bayes filter applied to Linear Gaussian Systems. Linear, Gaussian systems are defined by [10] to be systems whose states are modeled as Gaussian random variables, represented by the mean, $\boldsymbol{\mu}_{t_k}$, and the covariance matrix, $\mathbf{P}_{t_k}$. The probability of this Gaussian state vector, $\mathbf{x}_{t_k}$, is given as:

$$p(\mathbf{x}_{t_k}) = \det(2\pi\mathbf{P}_{t_k})^{-\frac{1}{2}} \exp\left\{-\frac{1}{2}(\mathbf{x}_{t_k} - \boldsymbol{\mu}_{t_k})^T \mathbf{P}_{t_k}^{-1}(\mathbf{x}_{t_k} - \boldsymbol{\mu}_{t_k})\right\} \tag{7}$$

Additionally, the state transition function is a linear function, represented in discrete time as:

$$\mathbf{x}_{t_k} = \mathbf{F}_{t_k}\mathbf{x}_{t_{k-1}} + \mathbf{B}_{t_k}\mathbf{u}_{t_k} + \mathbf{G}_{t_k}\mathbf{w}_{t_k} \tag{8}$$

where $\mathbf{F}_{t_k}$ is an $n \times n$ matrix known as the *state transition matrix.* $\mathbf{F}_{t_k}$ describes the evolution of the states being estimated between times t_{k-1} and t_k. $\mathbf{B}_{t_k}$ is the control matrix, sized $n \times m$ where m is the length of the control vector $\mathbf{u}_{t_k}$, and $\mathbf{G}_{t_k}$ is the $n \times n$ noise transformation/shaping matrix, and $\mathbf{w}_{t_k}$ is a matrix of zero-mean, white Gaussian noise, with covariance matrix $\mathbf{Q}_{t_k}$. Representing the mean of the state vector at time t_k as: $\boldsymbol{\mu}_{t_k} = \mathbf{F}_{t_k}\mathbf{x}_{t_{k-1}} + \mathbf{B}_{t_k}\mathbf{u}_{t_k}$, and substituting Equation (8) into (7) gives the state transition probability as Gaussian random variable:

$$p(\mathbf{x}_{t_k}|\mathbf{u}_{t_k}, \mathbf{x}_{t_{k-1}}) = \det(2\pi\mathbf{Q}_{t_k})^{-\frac{1}{2}} \exp\left\{-\frac{1}{2}(\mathbf{x}_{t_k} - \boldsymbol{\mu}_{t_k})^T \mathbf{Q}_{t_k}^{-1}(\mathbf{x}_{t_k} - \boldsymbol{\mu}_{t_k})\right\} \tag{9}$$

Assuming that the measurement probability, $p(\mathbf{z}_{t_k}|\mathbf{x}_{t_k})$, is also a linear function, subject to a zero-mean additive white Gaussian noise vector, $\mathbf{v}_{t_k}$, with covariance matrix, $\mathbf{R}_{t_k}$, where the measurement equation is defined by a linear system as:

$$\mathbf{z}_{t_k} = \mathbf{H}_{t_k}\mathbf{x}_{t_k} + \mathbf{v}_{t_k} \tag{10}$$

where $\mathbf{H}_{t_k}$ is a $k \times n$ (where k is the length of the measurement vector) sized matrix known as the *Measurement Matrix* which relates the state vector to the measurements provided by the sensor. The vector $\mathbf{v}_{t_k}$ represents a k sized vector of zero-mean, white Gaussian noise, with covariance matrix $\mathbf{R}_{t_k}$. The probability of the measurement conditioned on the state vector is a Gaussian PDF given by [10]:

$$p(\mathbf{z}_{t_k}|\mathbf{x}_{t_k}) = \det(2\pi\mathbf{R}_{t_k})^{-\frac{1}{2}} \exp\left\{-\frac{1}{2}(\mathbf{z}_{t_k} - \mathbf{H}_{t_k}\mathbf{x}_{t_k})^T\mathbf{R}_{t_k}^{-1}(\mathbf{z}_{t_k} - \mathbf{H}_{t_k}\mathbf{x}_{t_k})\right\} \tag{11}$$

Making the assumption that the prior term $p(\mathbf{x}_0)$ is a normally distributed random vector, with mean $\boldsymbol{\mu}_0$ and covariance matrix $\mathbf{P}_0$:

$$p(\mathbf{x}_0) = \det(2\pi\mathbf{P}_0)^{-\frac{1}{2}} \exp\left\{-\frac{1}{2}(\mathbf{x}_0 - \boldsymbol{\mu}_0)^T\mathbf{P}_0^{-1}(\mathbf{x}_0 - \boldsymbol{\mu}_0)\right\} \tag{12}$$

These assumptions allow the Kalman filter algorithm to maintain the representation of the posterior probability as a Gaussian random variable, the proof of which is given in [10].

The Kalman filter algorithm is conceptually divided into two steps. The first step is the *Prediction or Propagation Step.* This step calculates the belief of the state vector directly before incorporating the information from the measurement as shown in Equation (5). This is accomplished through use of the linear state transition equation outlined in Equation (8). The current covariance of the state estimate is also propagated through time.

$$\mathbf{x}_{t_k}^{-} = \mathbf{F}_{t_k}\mathbf{x}_{t_{k-1}} + \mathbf{B}_{t_k}\mathbf{u}_{t_k} \tag{13}$$

$$\mathbf{P}_{t_k}^{-} = \mathbf{F}_{t_k}\mathbf{P}_{t_{k-1}}\mathbf{F}_{t_k}^{T} + \mathbf{Q}_{t_k} \tag{14}$$

The next step in the Kalman filter algorithm is to incorporate information from the current measurement, known as the *Update Step.* The first part of the update step is the calculation of the *Kalman gain*, which is a weighting matrix which determines the amount of information from the measurement to include in the posterior belief through knowledge of the measurement covariance and the process noise. This matrix is calculated as:

$$\mathbf{K}_{t_k} = \mathbf{P}_{t_k}^{-}\mathbf{H}_{t_k}^{T}\left[\mathbf{H}_{t_k}\mathbf{P}_{t_k}^{-}\mathbf{H}_{t_k}^{T} + \mathbf{R}_{t_k}\right]^{-1} \tag{15}$$

The next part of the update step updates the mean of the state vector by incorporating the *innovation* or *residual*, which is the difference between the actual measurement from the sensor and the predicted measurement using the measurement:

$$\mathbf{r}_{t_k} = \mathbf{z}_{t_k} - \mathbf{H}_{t_k}\hat{\mathbf{x}}_{t_k}^{-} \tag{16}$$

The updated mean and covariance are then calculated as:

$$\mathbf{x}_{t_k}^{+} = \mathbf{x}_{t_k}^{-} + \mathbf{K}_{t_k}\mathbf{r}_{t_k} \tag{17}$$

$$\mathbf{P}_{t_k}^{+} = \mathbf{P}_{t_k}^{-} - \mathbf{K}_{t_k}\mathbf{H}_{t_k}\mathbf{P}_{t_k}^{-} \tag{18}$$

The derivation showing that these equations provide equivalence to the Minimum Mean Squared Error (MMSE) estimate of the state vector for linear Gaussian systems are provided both in [10] and [17].

2.3.1.2 Extended Kalman filter. The previous section outlined an implementation of the Bayes filter algorithm that was provably optimal in the MMSE sense for linear systems whose parameters are represented as a vector of Gaussian random variables. However, in many cases dealing with navigation, the state transition probability functions and the measurement probability functions may not be linear.

This non-linear state-transition function is defined as $\mathbf{f}$, whose parameters are the previous state estimate and current control inputs. The current state is propagated as:

$$\mathbf{x}_{t_k} = \mathbf{f}(\mathbf{x}_{t_{k-1}}, \mathbf{u}_{t_k}) + \mathbf{w}_{t_k} \tag{19}$$

where again $\mathbf{w}_{t_k}$ is a zero-mean additive white Gaussian noise vector of size n with covariance matrix, $\mathbf{Q}_{t_k}$. Similarly, the measurement function is generalized to a non-linear function $\mathbf{h}$ of the current state vector:

$$\mathbf{z}_{t_k} = \mathbf{h}(\mathbf{x}_{t_k}) + \mathbf{v}_k \tag{20}$$

where $\mathbf{v}_{t_k}$ is a zero-mean additive white Gaussian noise vector of size k with covariance matrix, $\mathbf{R}_{t_k}$. In the case of the Extended Kalman filter (EKF), an approximate linear model of Equation (19) is derived using a Taylor series expansion. The current estimate of the state is used to predict a nominal linearization point, $\hat{\mathbf{x}}_{t_k}$, and the partial derivative of the state transition function is used to model perturbations, $\Delta\mathbf{x}$, about that nominal point. The first-order Taylor series expansion is given as:

$$\mathbf{f}(\mathbf{x}_{t_k} + \Delta\mathbf{x}) \approx \mathbf{f}(\mathbf{x}_{t_k}, \mathbf{u}_{t_k}) + \mathbf{J}\left(\mathbf{x}_{t_k}\right)\Delta\mathbf{x} \tag{21}$$

where $\mathbf{J}_{t_k}$ is an $n \times n$ matrix of partial derivatives of the state transition function evaluated at the currently available estimate of the navigation state vector. This partial derivative matrix is known as the Jacobian of the state transition function, and is defined as:

$$\mathbf{J}_{t_k} = \left. \frac{\partial \mathbf{f}(\mathbf{x}, \mathbf{u})}{\partial \mathbf{x}} \right|_{\mathbf{x}=\hat{\boldsymbol{x}}_{t_k}, \mathbf{u}=\mathbf{u}_{t_k}} \tag{22}$$

Similarly, the first-order Taylor series expansion is used to approximate a linearized version of the measurement function in Equation (20), linearized about the current mean value of the navigation state vector:

$$\mathbf{h}(\mathbf{x}_{t_k} + \Delta\mathbf{x}) \approx \mathbf{h}(\boldsymbol{x}_{t_k}) + \mathbf{H}_{t_k}\Delta\mathbf{x} \tag{23}$$

where $\mathbf{H}_{t_k}$ is an $k \times n$ sized matrix of partial derivatives of the non-linear measurement function with respect to the current state estimate, known as the measurement Jacobian, and defined as:

$$\mathbf{H}_{t_k} = \left. \frac{\partial \mathbf{h}(\mathbf{x}_{t_k})}{\partial \mathbf{x}_{t_k}} \right|_{\mathbf{x}=\hat{\boldsymbol{x}}_{t_k}} \tag{24}$$

The Kalman filter steps can be reused to perform propagation and measurement updates, but with substitutions for the prediction parts of each step. For the propagation update, the prediction of the mean in Equation (13) is replaced with:

$$\hat{\mathbf{x}}_{t_k}^{-} = \mathbf{f}(\hat{\mathbf{x}}_{t_{k-1}}, \mathbf{u}_{t_k}) \tag{25}$$

Similar to the implementation from the Kalman filter in Equation (16), the calculation of the measurement residual for the EKF is given as:

$$\mathbf{r}_{t_k} = \mathbf{z}_{t_k} - \mathbf{h}_{t_k}(\hat{\mathbf{x}}_{t_k}^{-}) \tag{26}$$

The remaining steps for covariance prediction/update remain the same, substituting the Jacobian matrices for the original matrices in the Kalman filter equations in the previous section.

2.3.1.3 Histogram Filter. An alternative to parametric filters where the state estimate and measurement vectors are represented by Gaussian random variables are non-parametric filters. Non-parametric filters maintain an approximate estimate of the posterior over a finite decomposition of the state space. This research implements the *histogram filter*, which maintains a probability for each element of a discrete decomposition of the state space. A histogram filter where the discrete representation of the state space is mapped onto a metric grid, such as Spherical Mercator tiles, is referred to as a *grid filter*. A brief overview of the histogram filter is provided, where the full derivation can be found in [10].

In the histogram filter, the state vector, $\mathbf{x}_{\mathbf{t_k}}$, is decomposed into a series of K bins:

$$\mathrm{dom}(\mathbf{x}_{t_k}) = \mathbf{x}_{1,t_k} \cup \mathbf{x}_{2,t_k} \cup \ldots \mathbf{x}_{K,t_k} \tag{27}$$

where the function $\mathrm{dom}(\mathbf{x}_{t_k})$ defines the *domain*, or the set of possible values of the state vector, $\mathbf{x}_{t_k}$. Each bin, $\mathbf{x}_{k,t_k}$, represents a convex partition of the state space. For the histogram filter, each bin, $\mathbf{x}_{k,t_k}$, is assigned a probability, p_{k,t_k} that the k^{th} bin represents the true state. In the special case of the grid filter, the binning of the state space takes place over a uniform grid. The resulting posterior of the histogram filter maintains a uniform probability for each state x_{t_k} contained within each partition $\mathbf{x}_{k,t_k}$:

$$p(x_{t_k}) = \frac{p_{k,t_k}}{|\mathbf{x}_{k,t_k}|} \tag{28}$$

where $|\mathbf{x}_{k,t_k}|$ is the volume of the region of state space, $\mathbf{x}_{k,t_k}$. In the case of the grid filter, all regions occupy the same volume, and therefore can be factored out. Given a set of discrete prior probabilities, $p(\mathbf{x}_{t_{k-1}}) = \{p_{1,t-1}, \ldots, p_{K,t-1}\}$, the steps of the recursive Bayes filter are applied to the discrete state space. Using the control input vector, $\mathbf{u}_{t_k}$, the propagation step is applied:

$$p^{-}_{k,t_k} = \sum_i p\left(\mathbf{x}_{t_k} = \mathbf{x}_k \,|\mathbf{u}_{t_k},\, \mathbf{x}_{t_{k-1}} = \mathbf{x}_i\right) p_{i,t_{k-1}} \tag{29}$$

where the notation $\mathbf{x}_k$ indicates that the instance of the state $\mathbf{x}_k$ falls into the bin $\mathbf{x}_{k,t_k}$, e.g., $\mathbf{x}_k \in \mathbf{x}_{k,t_k}$. If the state vector is not truly discrete, or the propagation equations are implemented in a continuous case, an approximation is used to discretize the resulting probabilities. After the propagation step, the measurement vector is then used to apply the update step:

$$p_{k,t_k} = p(\mathbf{z}_{t_k}|\mathbf{x}_{t_k} = x_k)p^{-}_{k,t_k} \tag{30}$$

The collection of probabilities for each discrete state-space bin are then normalized such that:

$$\sum_{k=1}^{K} p_{k,t_k} = 1 \tag{31}$$

A full mathematical derivation of the histogram filter, and the discretization process for binning the state space is provided in [10].

2.4 Imaging

2.4.1 Introduction. This section will outline the modeling used to represent the EO camera sensor along with the image processing algorithms used as a basis for the image-aided navigation scheme outlined in the document.

2.4.2 Optical Sensor Model. A rigorous derivation of the optical imaging model is given in [18]. A review of the model is provided here, along with some additional information regarding image sampling and camera geometry. The camera/optical sensor is an analog to digital converter (ADC) which converts measurements of light intensity into a digital representation. The ADC itself is a collection of photon-sensitive elements, arranged in a 2-dimensional configuration known as a Focal Plane Array (FPA).

In this paper, the camera captures a 2-dimensional projection of the 3-dimensional *world*. As described in [18] the world consists of objects which are either sources of illumination or *radiance* that project that light onto other objects, or reflect light. The amount of light projected onto an object from a particular direction is known as *irradiance*. The irradiance pattern entering the camera from the world is known as the *scene* and is represented as a 2D projection onto a the FPA of the camera at time t, $\mathbf{o}(x, y, t)$. where (x, y) define a coordinate on the 2D plane corresponding to the FPA of the camera. To simplify the discussion the constant-illumination constraint is used, where the scene is assumed to be generated from a constant irradiance pattern being emitted from a piecewise-continuous 3D surface.

The light then enters the camera through the aperture and lens. The lens and aperture form a low-pass filter in the spatial domain, which is a function of the size of the aperture (D), the focal length of the lens f_0, and the wavelength of the light passing through the optical system λ. The optics of this system form the low-pass filter known as the Point-Spread Function (PSF), represented in the spatial domain as $\mathbf{h}(x, y)$ with a cutoff frequency of ν_c given by the relationship shown in [18]:

$$\nu_c = \frac{D}{\lambda f_0} \tag{32}$$

The image formed at the focal plane of the camera $\mathbf{i}(x, y, t)$ is a result of the 2-D convolution operation:

$$\mathbf{i}(x, y, t) = \int\limits_{\xi \in \mathbf{X}} \int\limits_{\rho \in \mathbf{Y}} \mathbf{o}(\xi, \rho, t)\mathbf{h}(x - \xi, y - \rho, t) d\rho d\xi \tag{33}$$

This operation can efficiently be performed in the spatial domain by multiplying the Fourier Transform of the scene $\mathbf{O}(\nu_x, \nu_y, t)$, with the Fourier transform of the PSF known as the *Optical Transfer Function* (OTF) $\mathbf{H}(\nu_x, \nu_y)$:

$$\mathbf{O}(\nu_x, \nu_y, t) = \mathcal{F}\{\mathbf{o}(x, y, t)\} \tag{34}$$

where the 2D Fourier transform is defined as [18]:

$$\mathcal{F}\{\mathbf{o}(x, y, t)\} = \iint \mathbf{o}(x, y, t) e^{-j2\pi(x\nu_x + y\nu_y)}\, d\nu_x\, d\nu_y \tag{35}$$

such that:

$$\mathcal{F}\{\mathbf{i}(x, y, t)\} = \mathbf{I}(\nu_x, \nu_y, t) = \mathbf{H}(\nu_x, \nu_y)\mathbf{O}(\nu_x, \nu_y, t) \tag{36}$$

Converting the image back to the spatial domain using the inverse Fourier transform provides the continuous image projected onto the focal plane array. At this point the image can be sampled by the pixel array. Given pixel sizes in each direction of $\Delta x, \Delta y$ the representation of the sampled image for pixel $[m, n]$ is given by:

$$\mathbf{i}(m, n, t_k) = \int\limits_{m-\Delta x/2}^{m+\Delta x/2} \int\limits_{n-\Delta y/2}^{n+\Delta y/2} \int\limits_{t_k-\Delta t/2}^{t_k+\Delta t/2} \mathbf{i}\left(x, y, t\right) dx\, dy\, dt \tag{37}$$

where Δt is the integration time of the camera in seconds. This provides a floating-point digital representation of the intensity of the image. Furthermore, aliasing problems can arise if the pixel spacing is insufficiently smaller than the Nyquist sampling rate determined by the OTF. The image is subject to two noise sources: read-off noise, and quantization error. Read-off noise is composed of ADC thermal noise and other conversion effects, and can be modeled as additive white Gaussian noise. Each sample is modeled as being corrupted by a sampled instance of a random variable drawn from a normal distribution with mean μ_r and variance σ_r^2 such that:

$$v_k(m, n, t_k) = \mathcal{N}(\mu_r, \sigma_r^2) \tag{38}$$

Each sample in the imaging system is converted to an integer value, and is additionally subject to quantization error based on the bit-depth of the imaging system. In this research, it is assumed that the magnitude of v_k is sufficiently dominant compared to the quantization error, such that quantization error can be ignored, and the final intensity value at pixel $[m, n]$ is given by:

$$\tilde{\mathbf{i}}(m, n, t_k) = \mathbf{i}(m, n, t_k) + v_k(m, n, t_k); \tag{39}$$

where the tilde denotes a noise-perturbed value of the true pixel intensity.

2.4.3 Projective Geometry. This section outlines a geometrical relationship for determining an intensity value at a discrete point on the camera focal plane array based on a model of camera optics and pose. The first issue is to develop the relationship between the various coordinate frame systems involved in the problem, using the conventions developed in Equation (1). For example, a rotation being applied to a vector in the camera frame, $\mathbf{t}^c$, in order to rotate it to the world frame, $\mathbf{t}^w$, would be described as:

$$\mathbf{t}^w = \mathbf{R}_c^w \mathbf{t}^c \tag{40}$$

The camera pose is given in terms of a rotation and a translation between the origin of the world frame and the center of the camera coordinate frame as $\mathbf{C}$, shown in Figure 3. The center of the camera $\mathbf{C}$ will be represented by a vector denoting the translation of the camera center represented in the world frame:

$$\mathbf{C} = \mathbf{t}^w \tag{41}$$

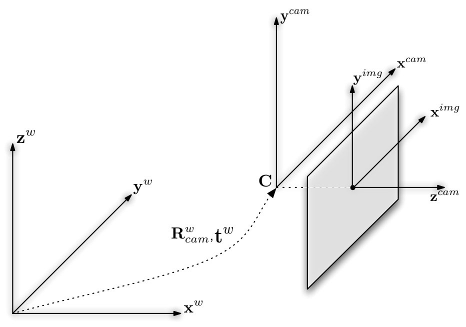


Figure 3: Coordinate Frames as Shown in [1]

The rest of this section defines the geometric component of the image rendering process. Under the constant-illumination constraint, the value of a pixel on the image plane is defined by the irradiance pattern of the scene, and the geometric relationship between the scene and the camera.

Typically, a 3D homogeneous representation or a point in space $\in \mathcal{R}(4)$ is employed, such that a 3D vector in inhomogeneous coordinates $\mathbf{x}_0^w = [x, y, z]^T$ would be represented in homogeneous coordinates as $\mathbf{X}^w = [wx, wy, wz, w]^T, w \neq 0$ [1]. To convert back to inhomogeneous coordinates, the w term is used to normalize the vector such that: $\mathbf{X}^w = [x, y, z, 1]^T$. The w coordinate serves as a regularization term for representing 3D points, commonly employed to aid numerical stability of points far away from the camera during various optimization routines.

The location of world points are localized in the camera frame by a perspective transformation. First, the rotation from the world to the camera frame $\mathbf{R}_w^{\text{cam}}$ is defined as the transpose of the rotation matrix between the world and camera frames:

$$\mathbf{R}_w^{\text{cam}} = [\mathbf{R}_{\text{cam}}^w]^T \tag{42}$$

As a matter of convenience, a vector $\mathbf{t}^{\text{cam}}$ describing the translation of the camera, in the camera frame, is defined as:

$$\mathbf{t}^{\text{cam}} = \mathbf{R}_w^{\text{cam}} \mathbf{t}^w \tag{43}$$

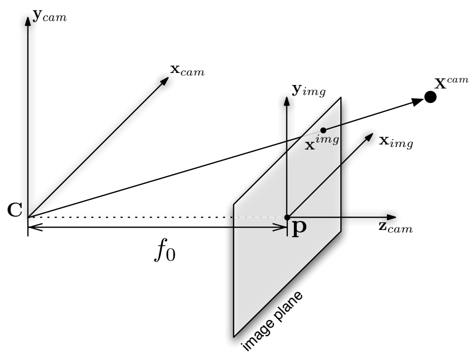


Figure 4: Camera and Image Frames as shown in [1]

The position of the discrete world points represented by the 3D vector $\mathbf{x}^w$ is computed in the camera frame:

$$\mathbf{x}^{\text{cam}} = [\mathbf{R}_w^{\text{cam}} | - \mathbf{t}^{\text{cam}}]\, \mathbf{x}^w \tag{44}$$

The pinhole camera model defines a linear relationship to project points located in the camera frame onto the image plane (the focal plane array), and is given by:

$$\mathbf{K} = \begin{bmatrix} f_0/\Delta x & 0 & p_x \\ 0 & f_0/\Delta y & p_y \\ 0 & 0 & 1 \end{bmatrix} \tag{45}$$

where f_0 is the focal length of the lens, Δx is the x dimension of a FPA pixel, Δy is the y dimension of a pixel. The principal point of the image plane is shown in Figure 4 defined as the point where the z-axis of the camera body frame intersects the image plane. This point is denoted as p_x, p_y and measured in pixels.

Points represented in the camera frame can then be projected onto the image plane using:

$$\mathbf{x}^{img} = \mathbf{K}\mathbf{x}^{\text{cam}} \tag{46}$$

Combining Equations (45) and (46) constructs the projection matrix $\mathbf{P}$, which yields pixel locations from discrete world points:

$$\mathbf{P} = \mathbf{K}\left[\mathbf{R}_w^{\text{cam}} | - \mathbf{t}^{\text{cam}}\right] \tag{47}$$

$$\mathbf{X}^{img} = \mathbf{P}\mathbf{X}^w \tag{48}$$

Notice that the pixel locations in Equation (48) are represented as 2D homogeneous vectors $\mathbf{X}^{img} = [x_i, y_i, z_i]$ where the actual pixel locations would need to be normalized by the z_i component of that vector.

This geometric relationship can also be used to back-project points on the image plane onto the scene. As the forward projection from the scene onto the image plane is a 3D to 2D projection, information about the relationship of the scene and the camera is lost in the projection process. The projection of image points onto the 3D scene is a process known as *ray-casting*, and is outlined here.

Given a point in the image frame, $\mathbf{x}^{img}$, ray-casting aims to determine the 3D world coordinates that correspond to the world point being imaged, $\mathbf{x}^w$. First, the inverse of the camera matrix is used to calculate the location of the image point, represented in the camera frame:

$$\mathbf{x}^{\text{cam}} = \mathbf{K}^{-1}\mathbf{x}^{img} \tag{49}$$

Given the camera center point in the world frame, t^w, the coordinates of the point on the image plane, represented in the world frame is then given as:

$$\mathbf{x}_0^w = \mathbf{R}_{\text{cam}}^w \mathbf{x}^{\text{cam}} \tag{50}$$

A ray is defined in the world frame, $\mathbf{r}_i^w$, which originates at the camera center, $\mathbf{t}^w$ and extends through the point $\mathbf{x}_0^w$ toward infinity. To recover the dimensionality lost when projecting from 3D to 2D, some assumption about the geometric structure of the scene must be used. A simple assumption that the scene is generated from a planar surface is a reasonable approximation for some of the algorithms in this research, and is presented here. To determine the point along $\mathbf{r}_i^w$ that intersects with the scene, $\mathbf{x}^w$, line-plane geometry is used to solve for the intersection point of the ray with the world ground plane. A plane is algebraically by the set of all points, $\mathbf{p}^w$, such that:

$$(\mathbf{p}^w - \mathbf{p}_0^w) \cdot \mathbf{n}^w = 0 \tag{51}$$

where $\mathbf{p}_0^w$ is a known point on the plane, and $\mathbf{n}^w$ is a unit vector normal to the plane. If the ray $\mathbf{r}_i^w$ is neither parallel to the ground plane, or lies completely within the ground plane, the intersection of the ray and the ground plane is the desired point, $\mathbf{x}^w$. Assuming the scene is represented by a plane in the x and y axes of the world coordinate system, the unit normal vector to the ground plane points in the z direction $\mathbf{n}^w = [0,0,1]^T$. The origin of the coordinate system is used as the known point on the plane, $\mathbf{p}_0^w = [0,0,0]^T$.

Given the center of the camera $\mathbf{t}^w$, and the ray, $\mathbf{r}_i^w$, defined as a unit vector pointing in the direction of $\mathbf{x}_0^w$, the point $\mathbf{x}^w$ is given as:

$$\mathbf{x}^w = \mathbf{t}^w + d\,\mathbf{r}_i^w \tag{52}$$

where d is a scalar representing distance along the ray. Substituting Equation (52) into Equation (51), and realizing that $\mathbf{p}_0^w$ is the zero vector, yields:

$$(\mathbf{t}^w + d\,\mathbf{r}_i^w) \cdot \mathbf{n}^w = 0 \tag{53}$$

Solving for d gives:

$$d = -\frac{\mathbf{t}^w \cdot \mathbf{n}^w}{\mathbf{r}_i^w \cdot \mathbf{n}^w} \tag{54}$$

2.4.4 Image Scale. This section defines the conceptual framework of image scale used in this research. Image scale is defined in two domains. First, the components of relative image scale are defined. Relative image scale is defined solely by the intrinsic properties of the camera. Secondly, using the projective relationship defined by the camera lens, and relative geometry to the scene, the relative scale can then be tied to an absolute, or global scale. A key concept of this research is the use of global scale to improve performance when comparing imagery from disparate cameras.

The relative scale of an image is defined by two components: inner and outer scale. *Inner scale* represents the size of an individual image element, or pixel. Inner scale is represented by $\Delta x, \Delta y$ in the pinhole camera model given in Equation (4). The *outer scale* of an image is the total area of the FPA. Outer scale normalized by the physical pixel size is known as the *resolution* of the image.

Absolute scale of the image uses the geometric relationship found in Section 2.4.3 to relate the relative scale of the image to the global coordinate system. For any given combination of camera model pose, the projection of the inner scale of the image onto the ground plane is known as the Ground Sample Distance (GSD). GSD measures the metric distance on the ground plane between two adjacent pixel centers. GSD is a scaling factor that can be used to convert the size of a pixel in an image to an absolute reference frame. GSD is not necessarily uniform across an image, and the uniformity depends on the pose of the camera relative to the ground plane.

Projecting the outer-scale of an image onto the ground-plane defines the Instantaneous Field Of View (IFOV). The IFOV represents the total amount of geographic area present in an image. For a 2D image with resolution (M, N), the GSD is related to IFOV by:

$$IFOV \approx GSD \times (M, N) \tag{55}$$

For any given image, there exists a range of scale values which any given element in the image can be generated from. This range, r, is defined as the effective scale range:

$$\{r \in \mathbb{R} : GSD \leq r \leq IFOV\} \tag{56}$$

This research creates a database of georegistered image keypoints and their descriptor vectors using high-resolution orthorectified imagery. Orthorectified imagery is a subset of georegistered imagery where the image is projected onto a local-level coordinate system such that the projection maintains equal GSD across the image. For many imaging applications, the effective scale range of the airborne platform will be a subset of the scale range of the database. This research uses the prior information from the airborne platform's navigation solution to compute the scale range and improve image processing performance.

2.4.5 Image Features. The majority of this research is focused on comparing imagery from an airborne camera to a set of reference imagery in order to determine aircraft position. This section derives a taxonomy of image features, which are statistics computed about an image used for computational tasks. This section also provides an overview of the image feature detectors and descriptors used in this research.

Generally defined [19], a *feature* is any deterministic function of the image $\mathbf{I}$, whose co-domain is the vector space $\mathbb{R}^K$, formally:

$$\boldsymbol{\phi} : \mathbf{I} \to \mathbb{R}^K \tag{57}$$

When using features to compare two sets of imagery, typically two features are employed: *feature detectors* and *descriptor extractors.* Feature detectors are designed to identify a *salient* portion of an image, or a region of the image whose intrinsic characteristics remain unchanged during a transformation of the original image. When using features for a task such as image matching, feature detectors that are *invariant* to various transformations are desirable. The output of an invariant feature detector is either not affected by a transformation of the image, or transformed similarly to the transform applied to the original image.

While in general, feature detectors yield any K-length vector, this research is concerned with the detection of *point-features*, corresponding to a single 2D point within the image. A point-feature detector yields a J-length vector called a *key-point*. A keypoint vector includes the 2D location of the point, and detector specific meta-data about the keypoint. For example, a point feature detector that finds the highest-intensity value of an image could be designed to return a 3-length vector: the x, y coordinates of the highest-intensity pixel in an image, and the intensity value at that point. A comprehensive review of feature detectors is provided in [20]. Common point-feature detectors include: Harris-Corners, the Scale Invariant Feature Transform (SIFT), and Features from Accelerated Segment Test (FAST) [21], [20]. The feature detectors used in this research are described in Chapter III.

Descriptor extractors are functions which generate K-length descriptor vectors, using either the entire image, or a sub-image local to a detected keypoint as input. In this research, descriptor vectors are typically used in discrimination tasks, and as such descriptor extractors are designed to generate descriptor vectors that minimize mutual information between vectors calculated from images generated by different scenes. One example of a descriptor extractor is a function that returns a fixed sized patch around a keypoint. Other common descriptor extractors are the Histogram of Oriented Gradients (HoG) used commonly with SIFT keypoints, or the binary vector descriptor Fast Retina Keypoint (FREAK) [22], [23]. This research uses compares three different descriptor extractors, detailed in Chapter III.

Computer vision literature commonly refers to 'features' as the combination of a keypoint and its associated descriptor vector. One additional concept used in this research is that of a *landmark*, which is a keypoint which has been projected into the world frame, and therefore has additional information about its 3D location. This dissertation aims to explicitly reference keypoints, descriptors, landmarks, and features, but may use these terms interchangeably.

2.5 Image-Aided Navigation

This section of the background will go into the basics of how navigation is performed. Specifically, this section will outline the basic framework for using an Inertial Measurement Unit (IMU) for computing a position, velocity, and attitude estimate, and using the recursive Bayes filter to estimate IMU error by incorporating measurements from external sensors.

2.5.1 Inertial Navigation. Inertial navigation is a form of dead reckoning, where given an initial estimate of position, velocity, and attitude with respect to a reference frame, the Inertial Navigation Solution (INS) uses measurements from accelerometers and gyroscopes to update its state over time. Errors in the accelerometers, gyroscopes, and gravity models contribute to a quadratic error rate on position over time. Accelerometer and gyroscope error models, along with the equations used to compute a strapdown navigation solution are covered in depth in [2] and summarized in [4] and will not be outlined here.

2.5.2 GPS Aiding of Inertial Navigation. Most current navigation systems incorporate some form of external aiding in order to bound the continuously drifting position error from the INS. GPS is a constellation of navigation satellites that provides a user with globally-available estimates of position and velocity. Multiple GPS receivers can provide a user with attitude estimates as well [24]. GPS estimates of user pose are typically drift-free; however these may be biased due to errors incurred from latent satellite state information, and insufficient modeling of signal atmospheric propagation. GPS receiver measurements are typically available at a lower rate than IMU measurements as well. Therefore, blending high rate yet drifting INS pose estimates with noisy GPS measurements allows for a smooth, drift free solution. There are many strategies on how to formulate the GPS measurements into an update for a recursive Bayesian estimation strategy, covered in [4] [10] [2] [24].

2.5.3 Simultaneous Localization and Mapping. Another popular strategy for reducing inertial drift is constraining vehicle pose by using observations of *landmarks* detected in the environment. This aiding type is commonly known as Simultaneous Localization and Mapping (SLAM) [10], where a Bayesian estimation strategy is employed to jointly estimate vehicle pose (and past poses) along with the locations of landmarks detected in the environment. Two main types of SLAM algorithms are currently employed for navigation and are outlined in the following subsections.

2.5.3.1 Visual Odometry. The first type of SLAM algorithm used in current vision-aided navigations systems is often referred to as Visual Odometry (VO) and has multiple implementations [4] [25]. In the VO scheme, landmarks are extracted from sensor measurements and then used over a short time period to constrain the estimates of pose. In this process, no long-term map is created, and the pose of the landmarks are marginalized out of the estimation process into the current pose estimate of the vehicle. VO is a case of dead-reckoning navigation, and the navigation state will still drift. However, this error in the VO system increases at a rate much smaller than a solution from an unaided strapdown inertial solution [4]. State of the art implementations of VO show performance on the order of less than one tenth of one percent of distance traveled [25].

2.5.3.2 Full SLAM/Perspective-n-Point. The "full" SLAM problem jointly estimates positions of landmarks along with keeping estimates of the entire trajectory of vehicle poses. VO techniques in [18] and [25] utilize the Extended Kalman filter, where error may be incurred through linearization about the current state estimate. In the EKF, this error is non-recoverable as all past information is marginalized into the current Gaussian representation of the state. In full SLAM a history is kept of observations, control inputs, and pose estimates. By keeping this history, error from linearization is recoverable through continuous re-linearization of the problem, at a cost of computational complexity. Full SLAM also tries to reduce navigation error though *loop-closure* which is the identification that a current observation was generated from a previously observed landmark. On loop-closure, the navigation state estimate error at the time of the original observation. Several techniques for online SLAM are discussed in [10], and recent advances in online bundle adjustment are documented in [26].

2.5.4 Image-Based Loop Closure/Appearance SLAM. Some current techniques for performing loop closure using a collection of images are outlined in the following sections.

2.5.4.1 Bag-of-Words Image Retrieval. Several systems have been developed for the purpose of identifying an image from a collection of images, a process called Content Based Image Retrieval (CBIR). The authors of [11] describe the use of text-retrieval techniques to identify images from a collection. In this scheme, a clustering algorithm such as k-means is used to cluster a collection of feature descriptor (SIFT) vectors into k representative groups. The average SIFT descriptor vector is used to represent each cluster, and is referred to as a visual-word. Then for each image in the collection, the image document can be represented by a histogram of visual word occurrences. Querying the collection is accomplished through quantizing the descriptor vectors from detected SIFT keypoints in the query image to their nearest visual words, and weighing the documents in the collection with some scoring metric. In [11] the authors use the *term-frequency, inverse-document frequency* or *tf-idf* heuristic. For each visual word in a document, tf-idf is computed using the frequency of words in a given document, down-weighted by the appearance of that word in the collection. The formula for this is given as [11]:

$$t_i = \frac{n_{id}}{n_d} \log \frac{N}{n_i} \tag{58}$$

where t_i is the tf-idf weight for word i, n_{id} is the number of times that word i appears in document d, n_d is the total number of unique words in document d, N is the number of documents in the collection, and n_i is the total number of times that word i appears in the entire collection. This weighting factor increases when words appear frequently in a document, but rarely in a collection. As a simple example, while the word "the" appears frequently in any given chapter of this thesis, the inverse-document term ($\log \frac{N}{n_i}$) down-weights the contribution for identification since "the" appears frequently in every document. The authors in [11] show good performance for retrieving images from movie clips given a query of a specific frame. Several approaches use this approach for place recognition or loop closure modules. Good examples are shown where this approach is used for the ability to determine if a mobile robot is revisiting a place in the map [27] or a single robot across multiple navigation sessions is visiting the same place [28].

2.5.4.2 FAB-MAP. A probabilistic approach to this strategy was developed in [12], called FAB-MAP. FAB-MAP is referred to as a *topological* SLAM method. Where most SLAM techniques operate in *metric* space, FAB-MAP models the world not by a geographical coordinate system, but as a set of distinct locations represented by their appearance in visual-bag-of-words space. In other words, FAB-MAP formulates the CBIR strategy into a recursive Bayesian estimation routine, maintaining a probability that the camera is either imaging a location that is already in the map, or a new location.

FAB-MAP composes an observation of a local scene at time t_k in image space by a binary vector, denoted as:

$$\mathbf{Z}_{t_k} = \left\{ z_1, \ldots, z_{|v|} \right\} \tag{59}$$

where each element z_i is equal to 1 if the word i was detected in the image and 0 if not observed, and v is the number of words in the vision-vocabulary. This observation model tries to model the existence of unobservable scene elements e_q such that given a feature detection / clustering model, the element z_i is modeled with a probability of missed detection / false alarm. The detector model is given as:

$$\mathcal{D}: \begin{cases} p(z_q = 1 | e_q = 0) & \text{probability of false-alarm} \\ p(z_q = 0 | e_q = 1) & \text{probability of missed-detection} \end{cases} \tag{60}$$

This model allows the incorporation of using a different detector model for different sensors / spectrum; however, that work is only theoretical and is left as future research. The map is composed of n_k locations, $\mathcal{L}^k = \{L_1, \ldots, L_{n_k}\}$, where the j^{th} location is modeled as:

$$L_j = \Big\{ p(e_1 = 1 | L_j), \ldots, p(e_{|v|} = 1 | L_j) \Big\} \tag{61}$$

The FAB-MAP algorithm also attempts to model correlation between visual words. The full joint probability between a large vocabulary of visual words would be very expensive to both store and computationally difficult to use in an online Bayesian inference. Therefore, the authors of [12] approximate the probabilities with a tree-based representation called the Chow-Liu tree to formulate the Bayes filter problem discussed in Equation (6). This formulation, given the model described above is shown as [12]:

$$p(L_i | \mathcal{Z}^{t_k}) = \frac{p(Z_{t_k} | L_i, \mathcal{Z}^{t_{k-1}}) p(L_i | \mathcal{Z}^{t_{k-1}})}{p(Z_{t_k} | \mathcal{Z}^{t_{k-1}})} \tag{62}$$

where $p(L_i | \mathcal{Z}^{t_{k-1}})$ represents the prior belief about the location given all past measurements $\mathcal{Z}^{t_{k-1}}$. The probability $p(Z_{t_k} | L_i, \mathcal{Z}^{t_{k-1}})$ is the observation likelihood, or the probability that the given observation was generated from the i^{th} location L_i, given all past observations. The probability $p(Z_{t_k} | \mathcal{Z}^{t_{k-1}})$ is a normalization term representing the probability of the current observation, given the history of all other observations. This normalization term is conceptually analogous to a "uniqueness" term. For the sake of brevity only the naive Bayes likelihood is explored here. In the naive Bayes formulation, the correlation between visual words are discarded. The full Chow-Liu tree model is described in detail in [12]. The observation likelihood for the naive Bayes formulation is given as:

$$p(Z_{t_k}|L_i) = \prod_{q=0}^{|v|} p(z_q|L_i) \tag{63}$$

where $p(z_q|L_i)$ is the probability that each element is generated for the current observation, given the model for location i. Noting that any joint probability between visual words is ignored, and and the detector model is constant over all locations yields:

$$p(z_q|L_i) = \sum_{s\in\{0,1\}} p(z_q|e_q = s)p(e_q = s|L_i) \tag{64}$$

where $p(z_q|e_q = s)$ is a function of the detector model and $p(e_q = s|L_i)$ is part of the location model. As this likelihood can be spiked, the authors derived a smoothing step.

The prior $p(L_i|\mathcal{Z}^{t_{k-1}})$ in this case is simply the last given location. The motion model applied between images in this case is simply a transition between sequentially mapped locations.

Finally, FAB-MAP introduces a normalization term, $p(Z_{t_k}|\mathcal{Z}^{t_{k-1}})$ which quantifies how unique a given observation is, given knowledge of all past observations and the location models. This term helps reduce the effect of perceptual-aliasing, or the generation of an observation with a high likelihood of being generated by several locations in the model. Depending on the scale of the problem, this term can either be generated from sampling the location models, or using some other representation of the world such as Google Street View.

FAB-MAP uses a threshold value to compare the result of Equation (62), $p(L_i|\mathcal{Z}^k)$, against to determine if the observation was generated by a location already in the map, or if it's a new location. If the algorithm determines that the observation was generated by a location already in the map, then the model for that location is updated with the information from the current update. Otherwise, a model for this location is initialized using both a model of a "mean" location, and the current observation update.

FAB-MAP 2.0 is developed as a scalable approximation of the FAB-MAP problem. Specifically several approximations are made for generation of the observation likelihood, and for updating location models [12].

2.5.4.3 Direct Feature Descriptor Matching. The most similar approach to the work presented in this dissertation is a direct feature descriptor matching approach [13]. This approach uses direct feature descriptor matching as a vote for a particular image in a sequence of Google Street View images. In this dissertation, the direct-matching approach is extended into a histogram-filter formulation, which enables the incorporation of other sensors to propagate our position belief between image updates. This allows the coarse-position estimation algorithm to maintain some performance even when operating in areas where the observation likelihood function does not provide sufficient information for localization , such as flying over featureless or non-distinct terrain.

III. Image Feature Keypoint Database

ACCURATE estimation of aircraft position using imagery is made possible through a continually updated model of the world's appearance. This Chapter provides insight into creation and maintenance of a discrete approximation of the world's appearance model, a database of georeferenced keypoints and their associated descriptor vectors.

3.1 Overview of Landmark Database

This chapter outlines the methodology and implementation details for the creation and refinement of a database of georegistered image keypoints and their associated descriptor vectors. This landmark database is then used in both the position acquisition and fine tracking algorithms described in Chapters IV and V, respectively. First, an overview of the various keypoint detectors and descriptor extractors used in this research is provided. A schema for representing the observed, georeferenced keypoints is derived. Then, this schema is used in a process for initializing a landmark database from a georeferenced satellite image. Next, a heuristic is derived to provide insight on the usefulness of any given landmark within the database for use in the navigation algorithms detailed in this research. Finally, this chapter shows how artifacts of the fine-tracking algorithm described in Chapter V can be used to refine the landmark database. Most of these sections provide details from an example implementation of the landmark database created from mosaic satellite image of Dugway Proving Grounds which was used as the reference database for the algorithms shown in Chapters IV and V.

3.2 Description of Keypoints and Descriptor Vectors

This research claims no contributions with respect to development of keypoint detectors or descriptor vector extractors; however, a brief outline behind the theory of some of the state-of-the-art keypoint detectors and descriptor vector extractors is given for context. An empirical evaluation of various combinations of keypoint detectors and descriptor vectors was performed using data described in Chapter V of this research, where airborne camera imagery was compared against satellite reference imagery. The results of the matching experiment led to the use of three keypoint detectors and their associated descriptor vectors in this research: The Scale Invariant Feature Transform (SIFT) [21], Speeded Up Robust Features (SURF) [29], and Binary Robust Invariant Scalable Keypoints (BRISK) [30]. The following sections provide an overview of the keypoint detectors and associated descriptor vectors used in this research.

3.2.1 Keypoint Detectors. An ideal keypoint detector is an image feature that can repeatedly identify a 2D point within an image, across any transformation of the image (scale, rotation, translation, affine, illumination) [30]. In practice, keypoint detectors will usually be robust to a subset of transformations of an image. In addition, detection performance usually comes at a cost of computation speed [31].

The first class of keypoint detectors use in this research use approximations of the second derivative of the image to identify points in the image with high frequency content. Both SIFT and SURF perform a second derivative operation across the scale decomposition of an image [32]. As the steps of the SURF keypoint detector are conceptually similar to the SIFT detector algorithm, the SIFT detector is detailed here.

In order to be robust to changes in scale, the SIFT keypoint detector first performs a scale decomposition of the image. The SIFT keypoint detector operates by locating extrema of the second derivate of the image by evaluating the smoothed Laplacian of the image. SIFT uses an approximation to the the Laplacian of Gaussian (LoG) kernel, the Difference of Gaussian (DoG) kernel. The DoG kernel is applied by first computing the Gaussian pyramid of the image, illustrated in Figure 5. Computing the Gaussian pyramid requires the convolution of the image with the Gaussian blur kernel defined as:

$$\mathbf{g}(x, y, \sigma) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} \tag{65}$$

Where the base image of the Gaussian pyramid, $\mathbf{l}(x, y, \sigma)$, is given as the convolution of the original image $\mathbf{i}(x, y)$ with the Gaussian blur kernel, $g(x, y, \sigma)$. This is represented in the spatial frequency domain as [4]:

$$\mathbf{L}(f_x, f_y, \sigma) = \mathbf{I}(f_x, f_y) * \mathbf{G}(f_x, f_y, \sigma) \tag{66}$$

where $*$ denotes element-wise multiplication between $\mathbf{I}$, and $\mathbf{G}$, the two-dimensional Fourier transform of the original image and the Gaussian blur function, respectively. Figure 5 shows the organization of the Gaussian pyramid. The base image is convolved with a series of Gaussian blur functions with increasing values of σ. Each doubling of σ composes an octave. The Gaussian pyramid decomposition is parameterized by s, the number of intermediate layers composing each octave. When the final image in each octave is computed, the blurred image $\mathbf{l}(x, y, 2\sigma)$ is downsampled by a factor of 2, and the next octave is computed.

Given the initial standard deviation used to blur the base of the Gaussian pyramid, σ_0, and the spacing between the layers of the pyramid, k, the i^{th} DoG image is computed by subtracting the intermediate layers of the Gaussian pyramid:

$$\mathbf{d}(x, y, i) = \mathbf{l}(x, y, k^{i+1}\sigma) - \mathbf{l}(x, y, k^{i}\sigma) \tag{67}$$

Once the difference of Gaussian images are computed, an extrema detection step searches for extrema in a 26 pixel region around each candidate pixel, spatially and in the two adjacent DoG images. Once a candidate pixel is identified, the Hessian of the image around that extrema is computed. The Harris metric, a function of the eigenvalues of the Hessian, is used as a threshold to reject unstable extrema [21]. A histogram of the orientation of the gradients about the image is computed, and the direction whose gradient has the largest magnitude is used to define the

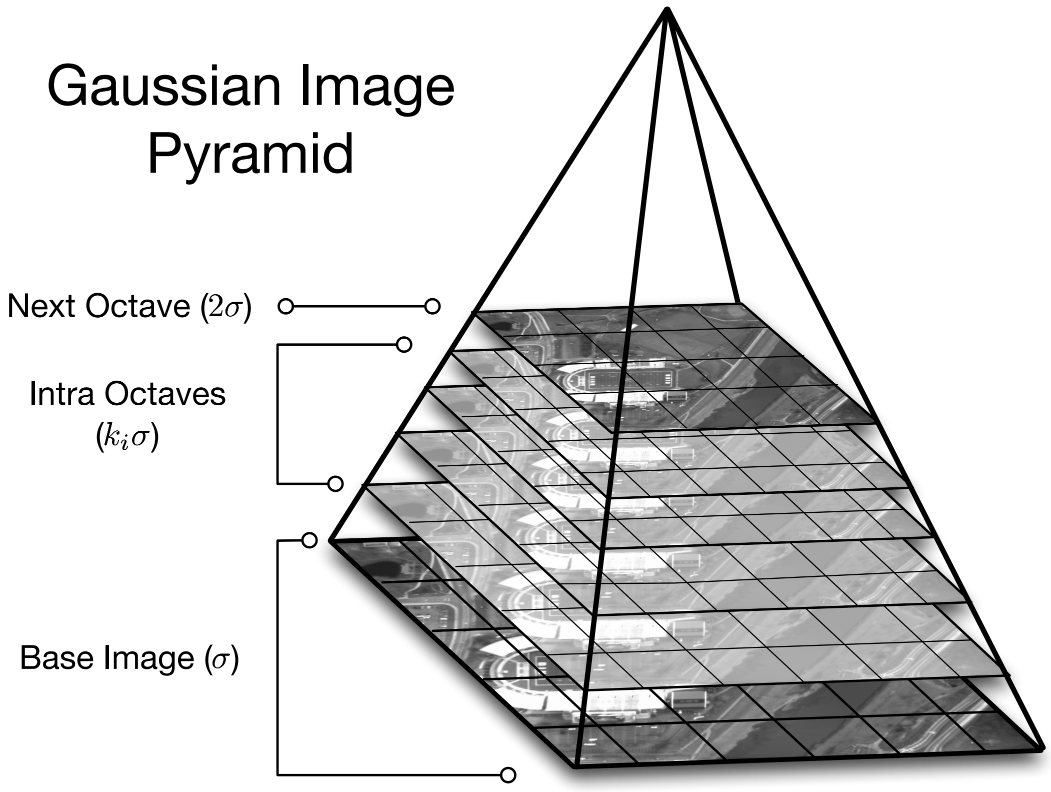


Figure 5: Scale-space decomposition of an image using the Gaussian Pyramid.

primary orientation of the keypoint. Normalizing the the keypoint about this reference orientation allows for the descriptor to be invariant to rotation. The j^{th}: keypoint returned from the SIFT detector is represented by a 5-length vector:

$$\mathbf{k}_j = \begin{bmatrix} x_j \\ y_j \\ \sigma_j \\ \theta_j \\ r_j \end{bmatrix} \tag{68}$$

where (x_j, y_j) is the two-dimensional location of the j^{th} keypoint in the base image, σ_j is the scale where the keypoint was detected, θ_j is the primary orientation of the j^th keypoint, and r_j is the Harris metric of the j^{th} keypoint, evaluated at (x_j, y_j, σ_j).

The SURF keypoint detector follows the same conceptual steps as SIFT, but with a different implementation to improve computation speed. SURF uses an approximate method to compute the Gaussian pyramid, based on square-shaped filters [29].

The BRISK keypoint detector combines some of the concepts behind SIFT, with the Features from Accelerated Segment Test (FAST) corner detector [31]. In the FAST-n algorithm, a candidate point $\mathbf{p}$ can be said to be a corner if a contiguous circle of n pixels around $\mathbf{p}$ all have higher intensity values than $\mathbf{p}$, or all have lower intensity values than $\mathbf{p}$. BRISK extends FAST by performing FAST keypoint detection on all scales of the Gaussian pyramid [30]. The BRISK detector locates keypoints by finding maxima of FAST scores across an image, and across neighboring scales. The final location of the keypoint is interpolated across both the image plane and scale space [30]. Estimation of the orientation of the BRISK keypoint is deferred until the descriptor vector is computed, at which point the keypoint vector from Equation (68) is returned.

3.2.2 Keypoint Descriptor Extractors. Once the keypoint is localized within the image, a descriptor vector extractor is then used to uniquely describe a local area around the keypoint. Descriptor extractors are decoupled from the keypoint detection process, so one could use the SIFT descriptor extractor with any of the keypoint detectors. However, the descriptor extractors are paired with their corresponding keypoint detectors for the purpose of this research.

The SIFT descriptor can be thought of a histogram of oriented gradients within a 16×16 pixel window, centered at the keypoint $\mathbf{k}_j$, aggregated into a 4×4 descriptor array. The orientations are binned into 8 directions, and then rotated about the nominal orientation of the keypoint, θ_j, to maintain rotation invariance [21]. This description process results in a $4 \times 4 \times 8 = 128$-length descriptor vector, $\mathbf{q}_j$. SIFT descriptor vectors are compared to each other using the L^2-norm [21]:

$$d_{uv} = \|\mathbf{q}_u - \mathbf{q}_v\|_2 \tag{69}$$

where d_{uv} is a scalar value denoting the L2 distance between the keypoint descriptor vectors $\mathbf{q}_u$ and $\mathbf{q}_v$. SURF calculates a similar descriptor, where the sum of Haar wavelets are sampled across a grid centered on a keypoint, resulting in a 64-length vector. The SURF descriptor vector also uses the L2-norm as a distance metric [29].

The BRISK descriptor also serves as a measure of the gradient around the keypoint, however is formulated as a 512-bit binary descriptor [30]. The BRISK descriptor uses a fixed sampling pattern around the keypoint, and stores the result of pairwise intensity comparisons between sample points as the descriptor. The resulting 64-byte binary descriptor vector is 16 times smaller than the SIFT descriptor, and 8 times as small as the SURF descriptor. Comparison between two binary descriptor vectors uses the Hamming distance, the sum of a logical exclusive-or operation (XOR) between two vectors. In addition to a reduction in descriptor vector size, computing BRISK descriptors is at least one order of magnitude faster than either SIFT or SURF, while matching is at least 3 times as fast [30].

The following sections make no assumptions about the choice of keypoint detector, or descriptor extractor. The schema for the keypoints remains constant across all the keypoint detectors, while the descriptor vector matrix varies only in the length of one dimension and the underlying data type when using binary descriptor vectors.

3.3 Landmark Database Schema

The landmark database is composed of two main components, the first being a table of the metadata returned from the keypoint detector, geographic metadata, and a heuristic metric of a landmarks suitability for airborne navigation. The other component is an $n \times m$ matrix of the n corresponding descriptor vectors, each of length m. The structure of the landmark database table, known as a schema, is presented here. Descriptions of how each of the fields are calculated are provided in subsequent sections.

- **Latitude**: WGS-84 latitude of the georeferenced keypoint (degrees).
- **Longitude**: WGS-84 longitude of the georeferenced keypoint (degrees).
- **Height**: Height above the WGS-84 ellipsoid (m).
- **NavRank**: Heuristically derived metric expressing suitability of landmark for use in online aircraft position estimation, derived in Section 3.5 (unitless, unsigned 8-bit integer).
- **Spherical Mercator Location**: Unique identifier for the Spherical Mercator tile that bounds the landmark. Used for improving database retrieval performance (unitless).
- **Absolute Scale**: Absolute scale of the landmark. Calculated by multiplying σ_j from Equation (68) by the GSD of the reference image (m)
- **Orientation**: Primary orientation of the landmark, θ_j, measured in degrees clockwise from true north (degrees).
- **Response**: Magnitude of the keypoint detection metric used in the detection step, given as: r_j, in Equation (68) (unitless).
- **Descriptor Link**: Mutable, implementation-specific reference to the descriptor vector currently associated with the landmark.

This schema can then be implemented using any underlying table-based analysis system which supports querying any of the fields described in the above schema. This research implemented the landmark database using the PyTables analysis-engine [33]; however this schema can easily be used by any modern relational database management system (RDBMS). In the implementation used by this research, the WGS-84 horizontal position, spherical Mercator location, detector response, and NavRank fields are indexed to support fast lookup into the database.

3.4 Initialization of Landmark Database

This section describes the creation of a landmark database from a mosaic of reference satellite imagery. The resulting database is composed of the landmark metadata table described above, along with the associated descriptor vector matrix.

3.4.1 Reference Imagery Coordinate Systems. The first step in creation of the landmark database it the composition of several coordinate system transformations which return the 3D WGS-84 position of a given 2D point in the reference imagery. The input to the database initialization process is a mosaic of georeferenced, orthorectified imagery. During the orthorectification process, the original image captured from the observation satellite or the airborne camera is corrected for perspective, such that the resulting image has a constant GSD over the entire imaivge extent [34]. During this process, the image is usually georeferenced, a process that results in the definition of a function that relates the coordinates of each pixel in the image to a position in a geographically meaningful coordinate system [35]. Figure 6 provides an illustration of the coordinate systems relevant to a georegistered image. The image frame origin is located in the upper left-hand corner of the image, with the x-axis pointing to the right of the image, and the y-axis pointing downward. In the case of an image aligned with true north, the x and y axes point east and south, respectively. The georegistered image metadata contains an affine transformation between the image frame, and a geographic coordinate system:

$$\mathbf{A}_{img}^{geo} = \begin{bmatrix} \Delta x & \theta_x & x_{ul}^{geo} \\ \theta_y & \Delta y & y_{ul}^{geo} \\ 0 & 0 & 1 \end{bmatrix} \tag{70}$$

where Δx and Δy represent the GSD of the image in the x and y directions of the geographic coordinate system, θ_x and θ_y account for the rotation between the image frame and the geographic coordinate system, and $[x_{ul}^{geo}, y_{ul}^{geo}]^T$ is the location of the upper left hand corner of the image in the geographic coordinate system. This affine transformation assumes that the image has been orthorectified, and projected onto a 2D coordinate system, resulting in a constant GSD across the image. Transforming a 2D pixel location $\mathbf{p}^{img} = [x^{img}, y^{img}]^T$ into geographic coordinates, $\mathbf{p}^{geo} = [x^{geo}, y^{geo}]^T$ is accomplished using the affine transformation:

$$\begin{bmatrix} x^{geo} \\ y^{geo} \\ 1 \end{bmatrix} = \begin{bmatrix} \Delta x & \theta_x & x_{ul}^{geo} \\ \theta_y & \Delta y & y_{ul}^{geo} \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x^{img} \\ y^{img} \\ 1 \end{bmatrix} \tag{71}$$

Converting a 2D position from the geographic coordinate system is accomplished using the inverse of $\mathbf{A}_{img}^{geo}$:

$$\begin{bmatrix} x^{img} \\ y^{img} \\ 1 \end{bmatrix} = \mathbf{A}_{img}^{geo\ -1} \begin{bmatrix} x^{geo} \\ y^{geo} \\ 1 \end{bmatrix} \tag{72}$$

The geographic coordinate system is a 2D coordinate system, defined by a (potentially non-linear) projection of the WGS-84 ellipsoid, $\mathbf{h}_w^{geo}()$, such that any point on the WGS-84 ellipsoid can be projected onto the specified 2D coordinate system by:

$$\mathbf{x}^{geo} = \mathbf{h}_w^{geo}(\mathbf{x}^w) \tag{73}$$

where $\mathbf{x}^w$ is a 2D vector consisting of WGS-84 latitude and longitude. Similarly, there exists the inverse projection function, $\mathbf{h}^w_{geo}()$, which projects points from the specified geographic coordinate system onto the WGS-84 ellipsoid.

The vertical position of a point located in the georegistered image is determined using an external Digital Elevation Model (DEM). Given a 2D position on the WGS-84 ellipsoid, the DEM provides an estimate of the height of the terrain in meters above Mean Sea Level (MSL). A model of the difference between MSL and the surface of the WGS-84 ellipsoid, known as *undulation*, is used to convert the height value from the DEM to height above the WGS-84 ellipsoid:

$$z^w = \mathbf{h}_{dem}(x^w, y^w) + \mathbf{h}_{geoid}(x^w, y^w) \tag{74}$$

where z^w is the calculated height above the WGS-84 ellipsoid, evaluated at the 2D WGS-84 coordinates, $\mathbf{x}^w = [x^w, y^w]$, $\mathbf{h}_{dem}$ is the non-linear function that returns MSL height at $\mathbf{x}^w$, and $\mathbf{h}_{geoid}$ returns the undulation value at $\mathbf{h}_{geoid}$.

This research uses the 1 arcsecond DEM compiled by the Shuttle Radar Topology Mission (SRTM) [36], along with the undulation model derived from the 1996 Earth Gravitational Model [37] to provide height estimates for points located within a georegistered image.

3.4.2 Creation of Landmark Database.

Once the composition of the transformation between the reference image frame and WGS-84 is completed, keypoints and their descriptor vectors are detected and extracted from the reference imagery. As the entire mosaic is too large to fit into memory for a typical desktop computer, the mosaic is split back into sub-images and landmarks are detected and described for each sub-image. The position of each keypoint in the image frame is then converted into the WGS-84 reference frame and stored in the landmark database along with the other metadata defined in Section 3.3. The NavRank heuristic is initialized

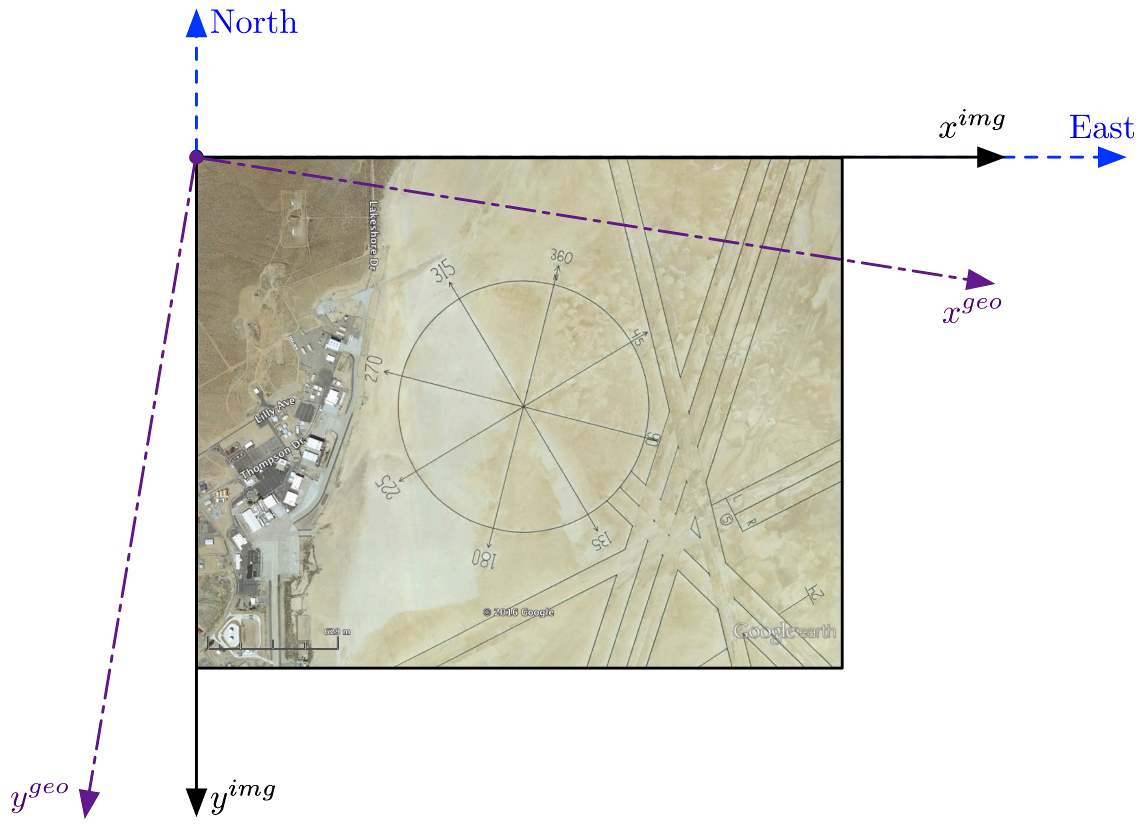


Figure 6: Illustration of the multiple coordinate systems relevant to a georegistered image. Georegistered imagery relates the origin of the image (upper left-hand corner) to a geographic coordinate system, which may be rotated about true north. Image of Edwards Air Force Base provided by Google.

to a constant value, and is later refined, as described in Sections 3.5 and 3.6. Each descriptor vector extracted is used as a row of the final descriptor vector matrix.

To validate the landmark database schema, 3 landmark databases were created, one for each of the BRISK, SIFT, and SURF keypoint detector and descriptor vector combinations. A roughly 45km × 55km mosaic of 0.6m GSD satellite imagery, a snapshot of which is shown in Figure 7, was used as input for each of these databases. For the PyTables implementation used in this research, both the landmark data table and the descriptor matrices are compressed using the Blosc compression algorithm [38].

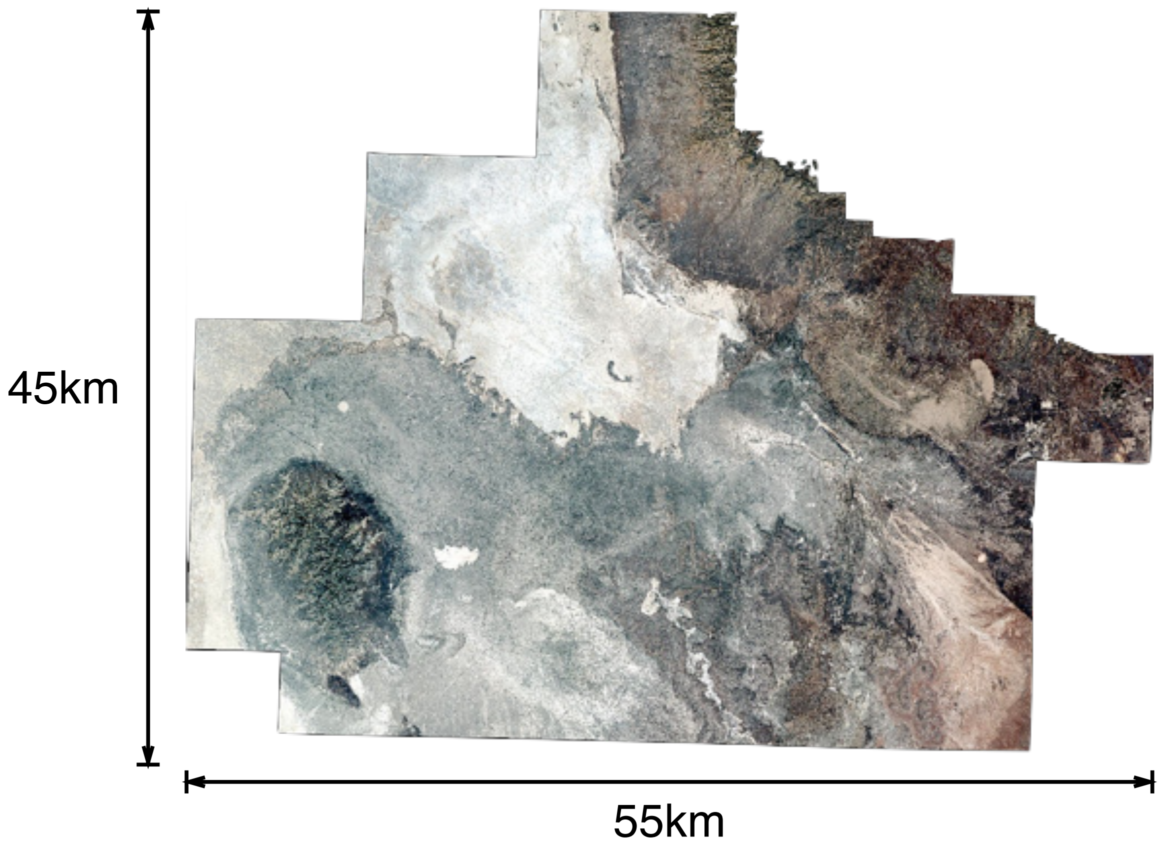


Figure 7: Overview of the orthorectified image mosaic used to initialize the landmark database. Image used under license through DigitalGlobe.

Table 1 shows the number of landmarks detected, the length of the descriptor vectors, and the final compressed size of the databases created for each keypoint detector and descriptor extractor combination. Table 1 shows that while the SURF descriptor vector is half the size of the SIFT vector, the SURF keypoint detector found roughly 2.6 million more landmarks, resulting in a database size similar to that of SIFT. A histogram showing the geographic distribution of the 32.25M landmarks detected and stored in the BRISK database is given in Figure 8, where the total number of landmarks, and their distribution were similar for the SIFT and SURF databases.

Table 1: Statistics of Landmark Databases Generated Using Multiple Keypoints

Keypoint	Number of Landmarks	Descriptor Size	Total Compressed Size
BRISK	35.25M	64 Bytes	3.34 GB
SIFT	32.76M	1024 Bytes	10.36 GB
SURF	35.36M	512 Bytes	11.92 GB

Once the database is initialized, the NavRank heuristic described in the next section is calculated for each landmark contained within the database. When NavRank is initialized, a search index is generated for the columns in the database corresponding to NavRank, WGS-84 horizontal position, Spherical Mercator tile, and absolute scale. These search indices allow efficient lookup during relatively complex queries into the database. The database can also be tuned for application-specific queries by pre-sorting the table, *e.g.,* first by Spherical Mercator location, and then by NavRank. This sorting allows for efficient queries of retrieving the N most useful landmarks for navigation in an approximate area. The next section derives the theory behind and implementation of the NavRank heuristic.

3.5 NavRank: Heuristic for Landmark Navigation Potential

The wide area position acquisition algorithm described in Chapter IV, and the fine-tracking algorithm derived in Chapter V both become computationally intractable as the number of reference landmarks used in each algorithm increases. When searching for nearest-neighbors in descriptor-vector appearance space, the probability of finding the true nearest-neighbor decreases as a function of the size of the search space [39].

With these concerns, both of the navigation algorithms developed in this research benefit from the ability to rate each of the landmarks in the database based on their usefulness in the algorithm. Several approaches use information-theoretic metrics to rank individual visual vocabulary words [40] [12], and the heuristic proposed in this research applies a similar methodology to direct-keypoint matching.

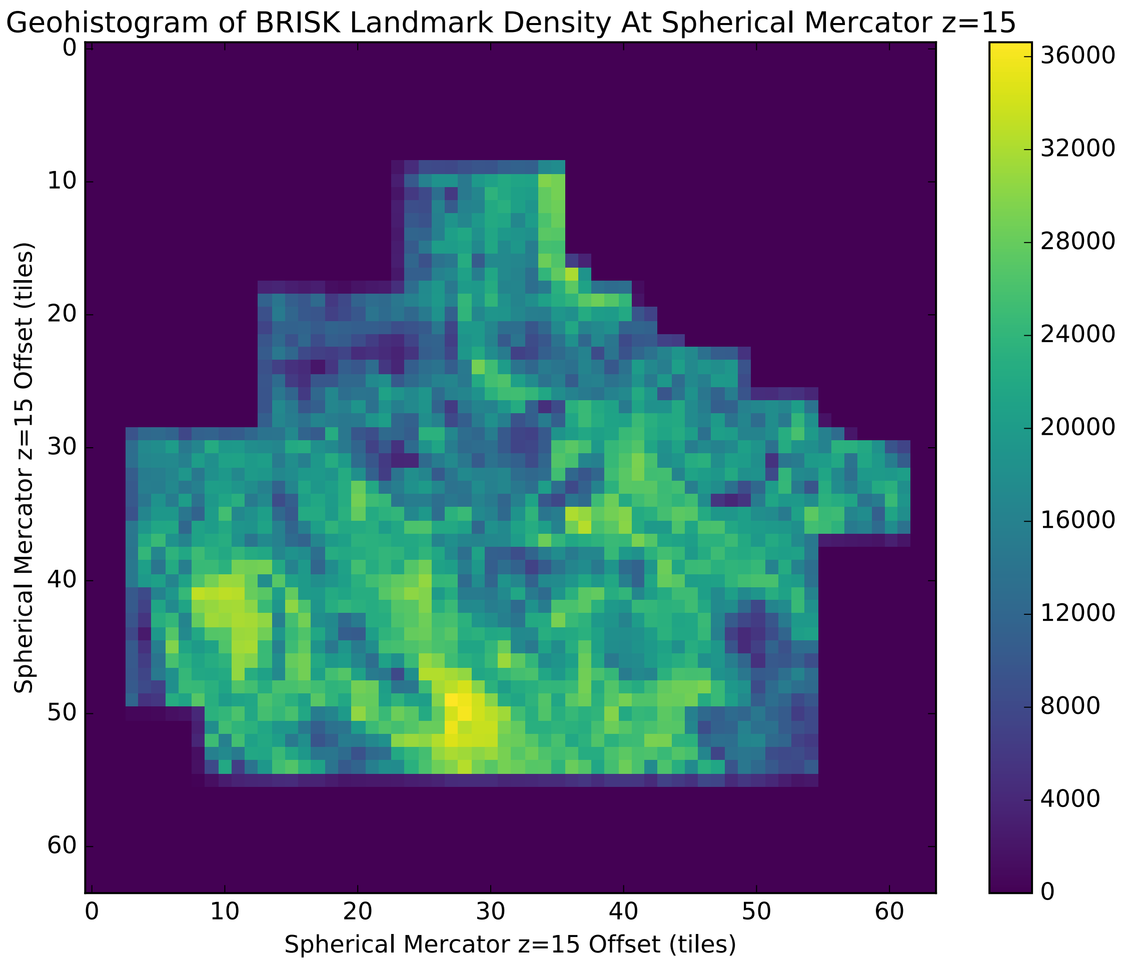


Figure 8: Geographic distribution of the 32.25M landmarks detected and described in the BRISK database, constructed from the Dugway orthophoto mosaic.

The heuristically derived metric for measuring the benefit of a particular landmark for use in an online navigation algorithm is named NavRank, for shorthand. When designing NavRank, intermediate results from the fine-tracking algorithm developed in Chapter V are used for statistical analysis. A brief overview of the fine-tracking algorithm and the intermediate results used in the development of NavRank is presented in the next section.

3.5.1 Landmark Classification. The fine-tracking algorithm described in this research is an implementation of a well-known algorithm, Perspective-n-Point (PnP), optimized for the flight environment. PnP formulates a non-linear least-squares optimization problem, estimating the parameters of camera translation and rotation with respect to a reference coordinate system that minimize the projection error of landmarks from the database onto the image plane of the camera at time t_k. A detailed description and analysis of the algorithm is given in Chapter V.

For the statistical analysis required to derive NavRank, landmarks from the database used in the PnP position estimation algorithm are classified into 4 categories depending how how an individual landmark is used within PnP. A computer science extension to the mathematical concept of a set, a multiset, is a set containing multiple instances of the same element [41]. An intermediate classification is the set of all the landmarks contained within the database, D. During one realization of the PnP algorithm, an *a-priori* estimate of aircraft position and attitude at time t_k is used to query the landmark database for a set of landmarks:

$$Q_{t_k} = \{v \mid v \in D, v \in \text{bbox}_{t_k}\} \tag{75}$$

where bbox_{t_k} is a geographic area that bounds the estimated ground footprint of the image observed by the airborne camera at time t_k. Given s observed images, the multiset Q is the union of all the queried landmarks, from time t_0 to time t_s:

$$Q = \bigcup_{k=0}^{s} Q_{t_k} \tag{76}$$

Next, the keypoint detector and descriptor vector extractor are applied to the observed airborne image at time t_k, yielding j keypoints and their associated vectors. The set of all keypoints detected and described from the image at time t_k is denoted J_{t_k}. For each observed keypoint $u \in J_{t_k}$, the two landmarks returned from the database query with the smallest distance to the observed keypoint descriptor vector,

v_0 and v_1 are determined using exhaustive search. The distance between keypoint u and v is denoted d_{uv}, and computed either using Hamming distance in the case of binary descriptor vectors, or the L2-norm when using SIFT or SURF descriptor vectors.

The set C_{t_k} denotes correspondence candidates between the J_{t_k}, set Q_{t_k}. This set is defined by comparing the ratio of the two closest descriptor vectors, $s_{uv_0} = d_{uv_0}/d_{uv_1}$ to an implementation-specific threshold, τ:

$$C_{t_k} = \{(u, v) \mid s_{uv} < \tau, \quad u \in J_{t_k}, v \in Q_{t_k}\} \tag{77}$$

The correspondence candidates at time t_k are then used by the PnP algorithm to compute an estimate of aircraft position. The outlier detection method detailed in Section 5.2.1 is then used to compute the set I_{t_k}, which is the set of landmarks from correspondence candidates that were determined to be inliers, and actually used in the pose estimation problem. While the detailed derivation of the outlier detection function is postponed until Section 5.2.1, it is represented here as a scalar function $o(u, v)$, that returns 1 if the correspondence pair (u, v) was determined to be an outlier, and 0 otherwise. The set of inliers at time t_k is then given as:

$$I_{t_k} = \{v \mid o(u, v) = 0, \quad (u, v) \in C_{t_k}\} \tag{78}$$

where the complement to this set, O_{t_k}, is the set of landmarks from correspondences candidates that the PnP outlier detection algorithm determined to be outliers:

$$O_{t_k} = \{v \mid o(u, v) = 1, \quad (u, v) \in C_{t_k}\} \tag{79}$$

These sets of inlier and outlier landmarks are then aggregated over the s observations to form the multisets I and O. A *flight* is defined as the collection of s

observations from time t_0 to t_s. The resulting sets and multisets are then used to define the following 4 categories:

- **Inlier Landmarks**: The multiset I of all landmarks that were used as inliers during the flight.
- **Outlier Landmarks**: The multiset O of all landmarks that were determined to be outliers during the flight.
- **Queried Landmarks**: The multiset Q of all landmarks that were retrieved from the database during the flight. Conceptually, this is the multiset of landmarks that were 'seen' during the flight.
- **Unused Landmarks**: The set U, defined as the difference between D and Q. Conceptually, the set of landmarks that were never queried from the database.

The following Sections show two analyses of the landmark database, using these four categories as conditional variables to provide insight into the design of NavRank. The first analysis was designed to identify any intrinsic properties of the data that could be used to predict which landmarks would be grouped into the inlier landmark multiset I. The second analysis would use statistics on the 4 (multi)sets themselves to predict that future observations of a landmark would result in the landmark being classified as an inlier.

These analyses drew data from all 4 flights conducted during the Dugway flight test campaign, described in Chapter V. This analysis used the BRISK keypoint detector and descriptor vector. An earlier analysis was performed using SIFT providing similar results, drawing the same resulting conclusion as the analysis discussed in the next Section. The number of the landmarks classified into each category is shown in Figure 9. The unused landmarks for the BRISK landmark database form the largest category, not pictured in Figure 9, consisting of roughly 32.85M landmarks. The category of landmarks that were queried, and then never considered as a component in a correspondence candidate form the next largest category at 3.25M landmarks. The

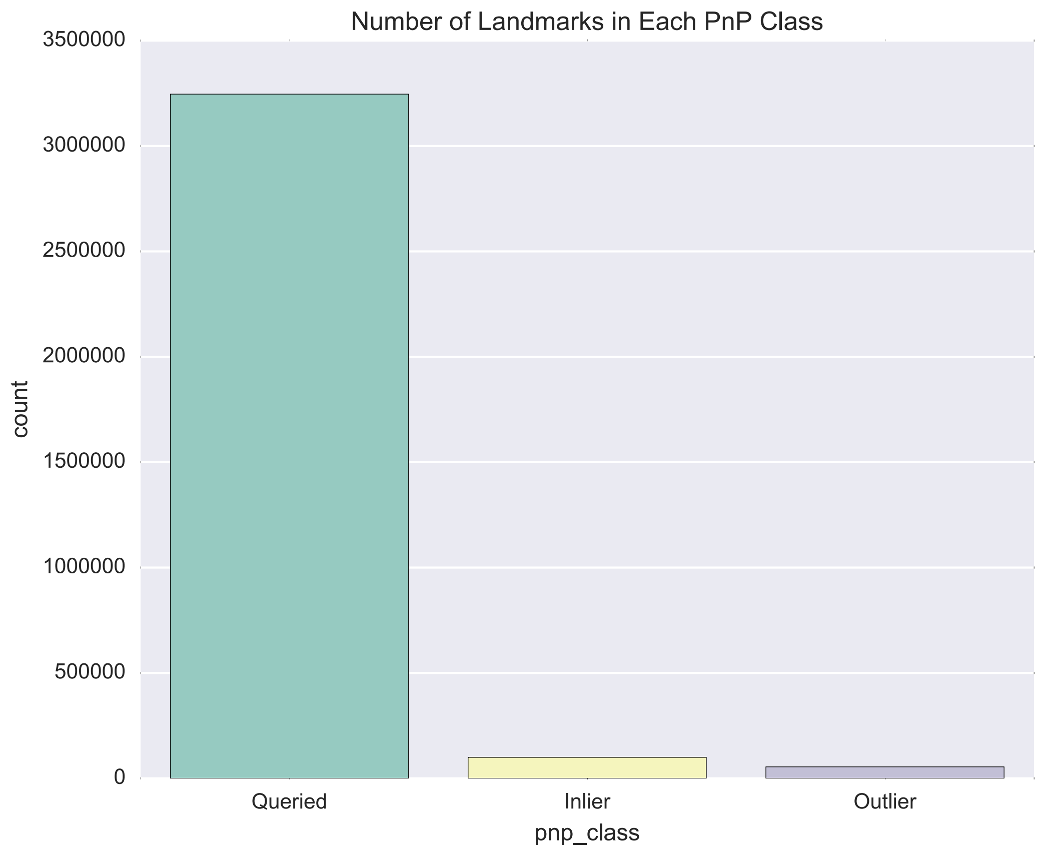


Figure 9: Number of landmarks classified into three of the PnP classes. The unused landmarks category compose the largest category, containing 31.85M landmarks.

categorical analysis shows that roughly 100K of the landmarks were used as inliers, and 54k of the landmarks were consistently identified as outliers.

3.5.2 Landmark Database Analysis. The intrinsic analysis of the landmark database aims to identify any distributions of the separate components of the keypoint vectors which are correlated to their classifications as inliers or outliers. The identification of any intrinsic property of they keypoint vectors which is positively correlated to being used as an inlier in the PnP algorithm could be used as a stand-alone component of the NavRank heuristic.

During the database analyses, a conditional probability distribution function for three fields of the landmark database schema (keypoint primary orientation, keypoint size, and keypoint detector response) were estimated, conditioned on the classification of each landmark. The analysis conditioned the distributions on the three observed categories: inliers, outliers, and queried. The analysis disregards the unused landmark classification.

Initial analysis was performed visually, by constructing kernel-density estimates of the conditional probability distribution functions. The kernel-density estimates are illustrated in Figures 10-12 as violin plots, which are a hybrid between a traditional box-and-whiskers plot, and a histogram. The outer shape of the violin in Figures 10-12 represents the median value, the inner dark line represents the inner 2 quartiles (50% of the data), where the colored-in area represents the entire dataset. Extended lines in the violin-plots represent outliers, or connections between two modes of the distribution.

Visually inspecting the distributions results in the identification that only keypoint detector response shows any difference between the conditional distributions for each category. A more traditional boxplot of keypoint detector response is shown in Figure 13. This result shows that landmarks that were used as inliers in the PnP pose estimation process contain keypoints with a stronger response value, on average. This result was used in the NavRank development process as a rule to prioritize landmarks with a higher response value.

3.5.3 Derivation and Implementation of NavRank. While the intrinsic analysis shows that the response value of a keypoint can be used to select a subset of the landmarks from the database that contain inliers, the resulting query still contains outliers along with a large number of unused landmarks. Using the multiset information from the classification process can then be used to identify landmarks which are consistently used as inliers in the PnP algorithm. However, insight from the classification process only aids in ranking landmarks previously observed, Q, and provide no additional insight into U.

The principles behind the NavRank heuristic are defined as follows:

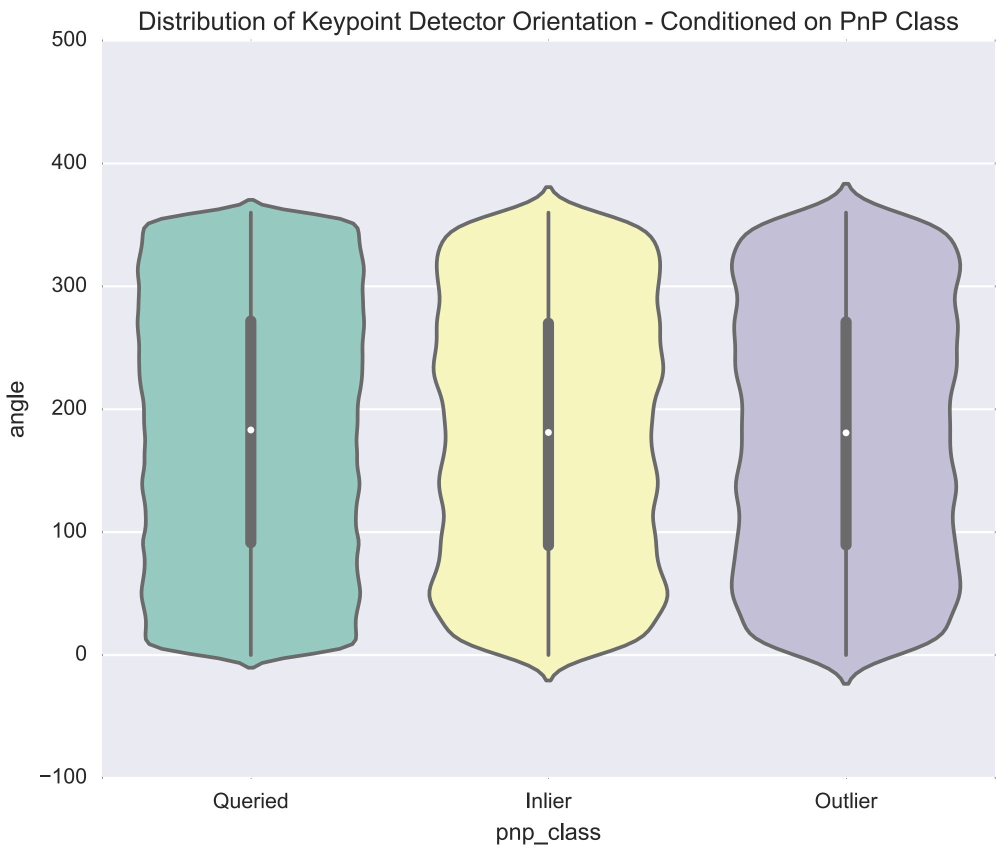


Figure 10: Violin plot, showing kernel density estimates of the probability distribution function of the primary orientation of the landmark keypoint (degrees), conditioned on the classification of the landmark by its use in the PnP algorithm.

- Prioritize landmarks consistently used as inliers in the PnP algorithm, I. Use the ratio between the multiplicity of a landmarks appearance in I, to the number of times the landmark was queried, its multiplicity in Q as the within-group ranking metric.
- Demote landmarks which are consistently identified as outliers, O. By definition, the multiplicity of a landmark in O is equal to its multiplicity in Q. Conceptually, these landmarks consistently pass the appearance-matching stage and are identified as correspondence candidates, but are always rejected by the outlier-detection step in PnP.

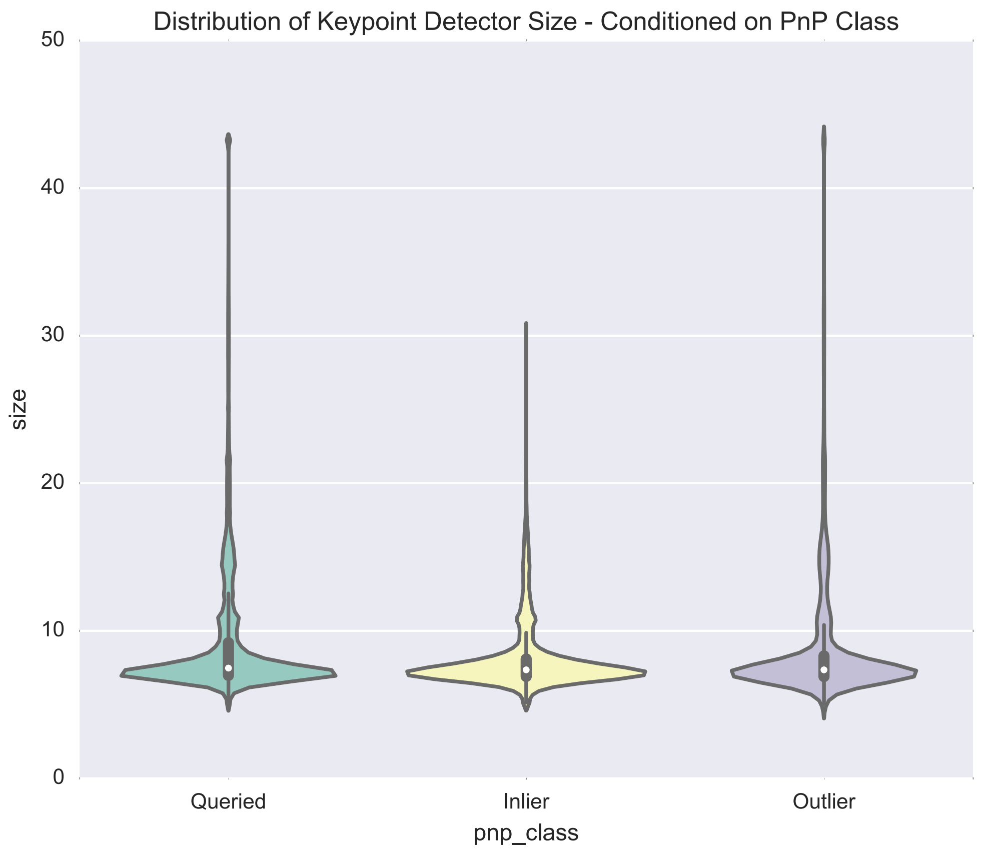


Figure 11: Violin plot, showing kernel density estimates of the probability distribution function of keypoint detector size neighborhood (meters), conditioned on the classification of the landmark by its use in the PnP algorithm.

- Rank all other landmarks in D not in O or I in-between I and O, prioritized by their keypoint detector response value.

These concepts are used to derive the NavRank heuristic, an illustration of which is shown in Figure 14. The NavRank heuristic is implemented as a scalar value for each row in the landmark database. Each landmark classified as an outlier is assigned some minimal value, τ_{r0}. Landmarks classified as inliers are assigned a value between τ_{r2} and the implementation-specific maximum value, τ_{max}, ranked by the ratio of the number of times each landmark was used as an inlier to the number of times it was queried. The landmarks not classified as inliers or outliers, $D \setminus (I \cup O)$, are assigned a value between τ_{r1} and τ_{r2}, sorted in increasing order of landmark keypoint response.

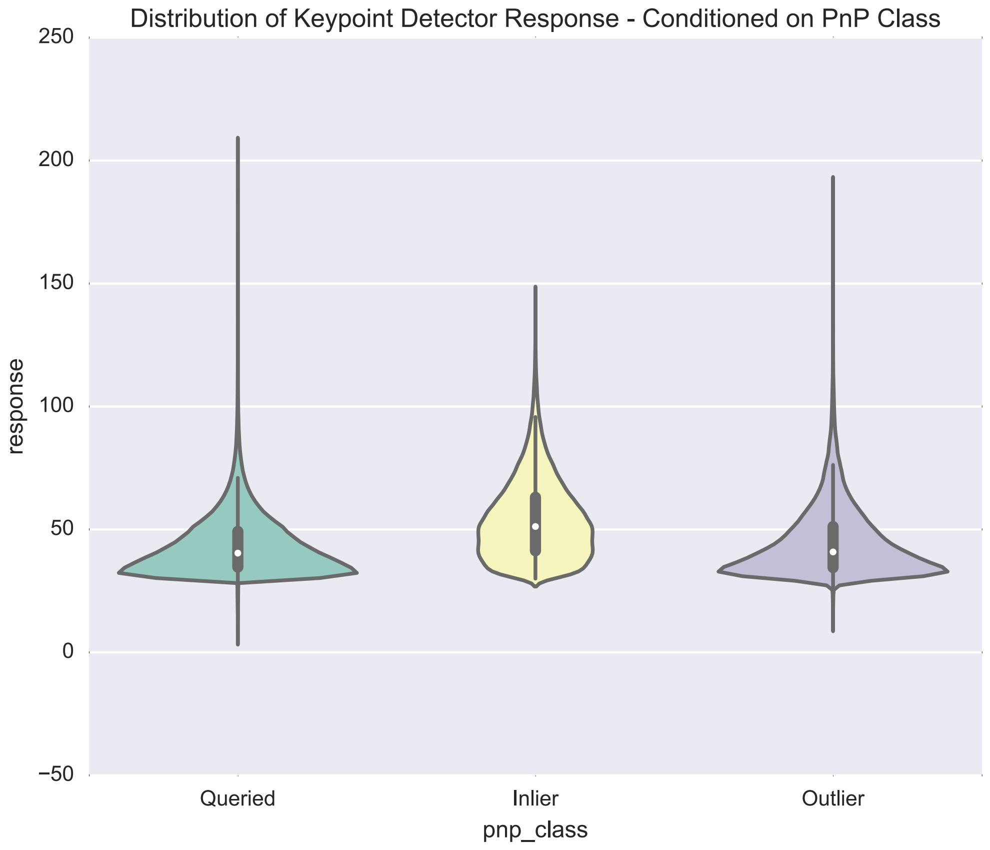


Figure 12: Violin plot, showing kernel density estimates of the probability distribution function of keypoint detector response, conditioned on the classification of the landmark by its use in the PnP algorithm.

The NavRank metric for the i^{th} landmark, $\mathbf{l_i}$ is denoted as τ_i and formalized by:

$$\tau_i = \begin{cases} \tau_{r0} & \text{when } \mathbf{l_i} \in O \\ \dfrac{r_i}{r_{max}}\left(\tau_{r2} - \tau_{r1}\right) + \tau_{r1} & \text{when } \mathbf{l_i} \in D \setminus (I \cup O) \\ \dfrac{m(\mathbf{l_i}, I)}{m(\mathbf{l_i}, Q)}\left(\tau_{max} - \tau_{r2}\right) + \tau_{r2} & \text{when } \mathbf{l_i} \in I \end{cases} \tag{80}$$

where r_i is the keypoint detector response for landmark $\mathbf{l_i}$, r_{max} is the maximum value of all response values in the database, and $m(\mathbf{l_i}, M)$ is the multiplicity function, which returns the number of times a landmark occurs in the multiset, M.

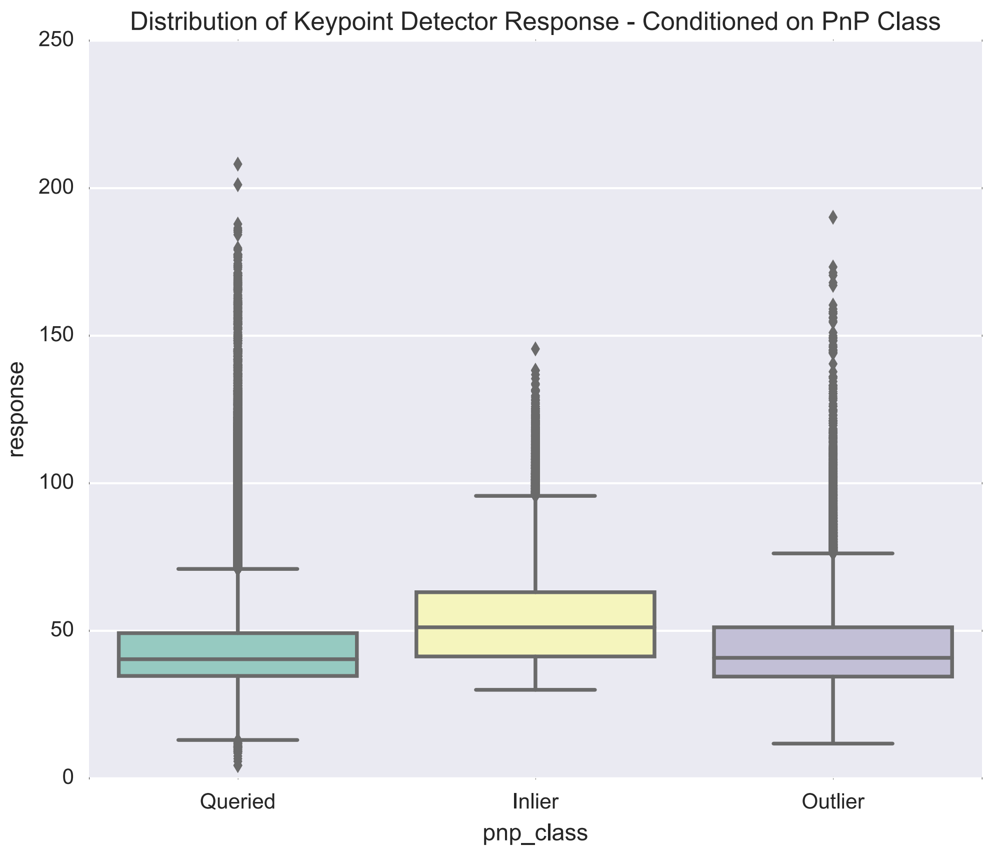


Figure 13: Box plot illustrating quartiles of the distribution of keypoint detector response, conditioned on the classification of the landmark by its use in the PnP algorithm.

The PnP algorithm was then used in an experiment to validate the NavRank metric. In the experiment, the first 3 flights from the Dugway flight test campaign described in Chapter V were used to create landmark classifications. The landmark classifications where then used to compute the NavRank metric for each landmark in the database. Three derivative landmark databases were created by subsampling the original database using varying thresholds of NavRank. The first database, $L0$, was a direct copy of the original. The second, $L1$, consisted only of landmarks that were previously used as inliers in flights 1-3 by selecting all landmarks with a NavRank value greater than τ_{r2}. The third database, $L2$, used a threshold τ_w, resulting in a database of the top 10% of landmarks according to NavRank.

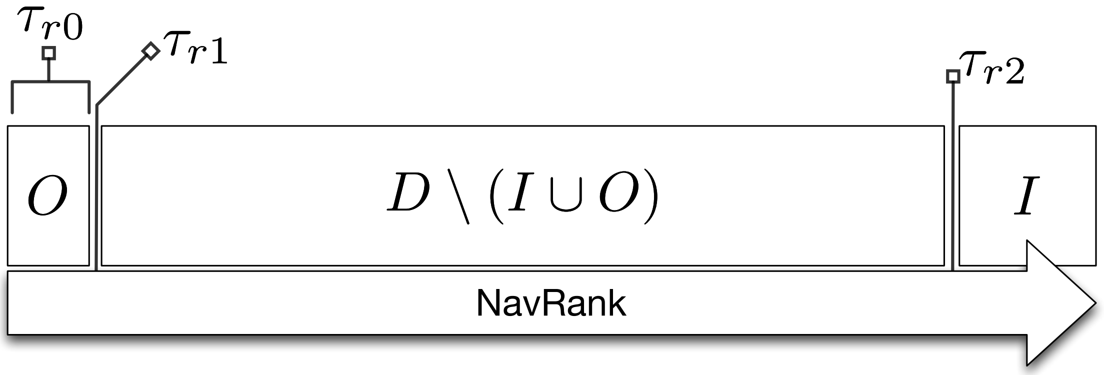


Figure 14: Illustration of the NavRank metric. NavRank is used as a landmark database selection criteria when querying the database for a maximum number of landmarks. NavRank places the least priority on outliers O, highest priority on inliers I, ranked by the ratio of times a landmark is used as in inlier to the number of times it was queried from the database, and ranks the remaining landmarks in the database $(D \setminus (I \cup O))$ as a function of their keypoint response value.

The PnP algorithm was then used to estimate the position of the aircraft for Flight 5, conditioned on the use of each database: $L0, L1, L2$. The number of observations which generated a valid PnP result is illustrated in Figure 15. The control database, $L0$, generated the most PnP position estimates at 770. Impressively, when using database $L2$, PnP generated nearly the same number of position estimates (736), with a database only 10% as large as $L0$. Additionally, using a database consisting of only previously observed inliers, $L1$, PnP was still able to generate 396 estimates of aircraft position. $L1$ consists of roughly 100K landmarks, approximately 0.30% the size of $L0$.

The 3-dimensional RSS error of the 3 degree-of-freedom PnP algorithm, derived in Chapter V, is presented in Figure 16. When position estimates were generated, the resulting accuracy was similar across the ensemble. Interestingly, while the PnP algorithm using the $L1$ database generated a position estimate roughly only half as often as $L0$ or $L2$, it generates no significant outliers. Overall, this experiment shows that implementation of a heuristic that uses a combination of intrinsic keypoint informa-

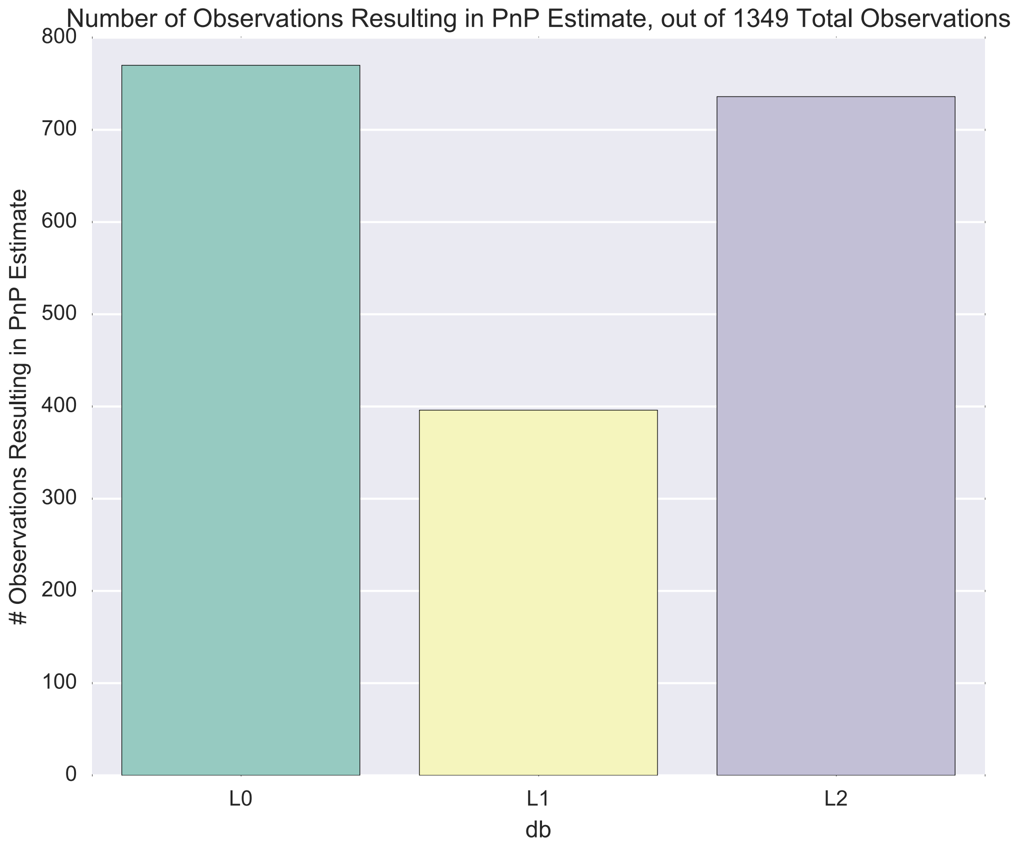


Figure 15: Number of observations resulting in a valid position estimate from PnP for Flight 5, using BRISK, conditioned on databases created using different NavRank thresholds. Database $L0$ is the control database. $L1$ uses a database of only previously observed inliers. $L2$ is a database created using the top 10% of landmarks in $L0$, according to NavRank.

tion along with statistics of previous observations can be used to maintain navigation performance while significantly down-sizing the size of the reference database needed.

3.6 Continuous Refinement of Landmark Database

The last section demonstrated that new observations of landmarks contained within a database can then be used to predict which landmarks will be useful in the future. The information gained from these new observations is heuristically marginal-

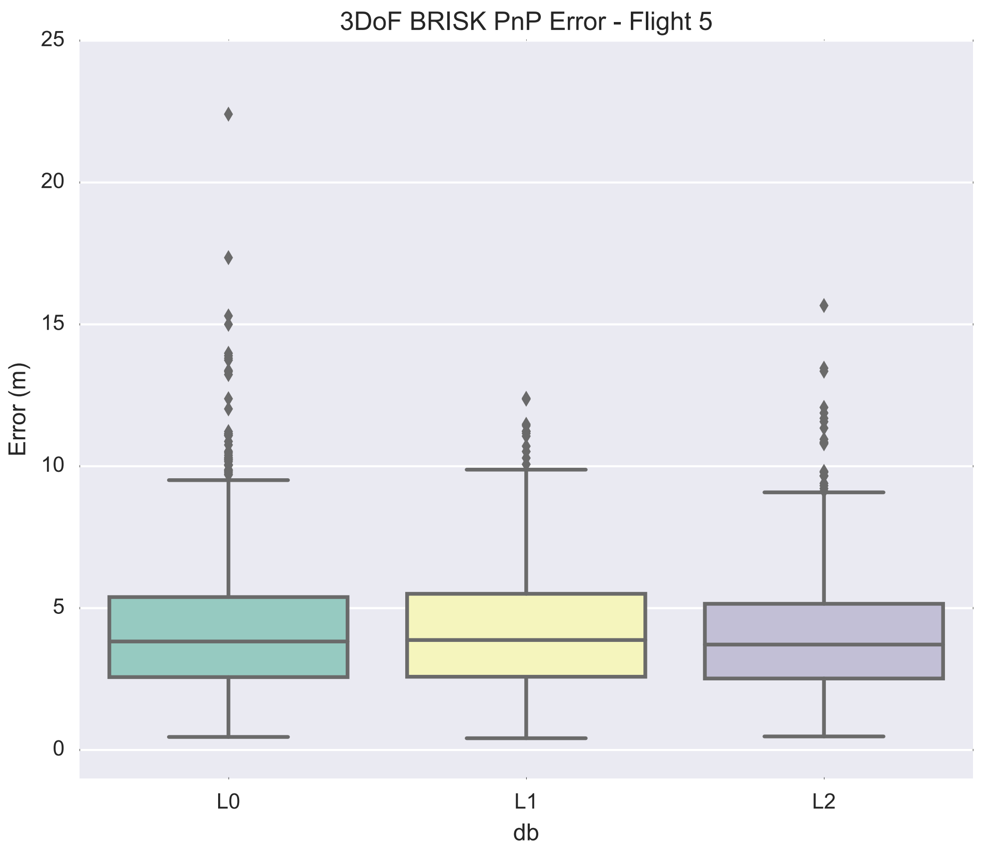


Figure 16: 3D RSS error of the PnP position estimation algorithm for Flight 5, using BRISK, conditioned on databases created using different NavRank thresholds. Database $L0$ is the control database. $L1$ uses a database of only previously observed inliers. $L2$ is a database created using the top 10% of landmarks in $L0$, according to NavRank.

ized into the NavRank metric. Updating NavRank can be accomplished incrementally when a new set of observations is available.

3.6.1 Landmark Position Update. In addition to updating NavRank, the other landmark data can also be updated after additional observations. For example, Section 5.3 outlines a batch estimation problem that uses airborne observations of the landmark database to estimate a global bias in the 3D WGS-84 position of the landmarks. This bias is subtracted from the WGS-84 position of each landmark.

Section 5.5.1 shows the improvement in PnP accuracy, given the ability to incorporate additional information about landmark position.

3.7 Chapter Summary

This chapter presented a method for creating a database of georegistered keypoints from a mosaic of reference satellite imagery. To create this database, a schema that describes the contents of the database was derived, and validated using three different keypoint detectors and descriptor vectors. To validate the database creation process, several landmark databases were generated from georegistered satellite orthophotos. A heuristically derived metric, NavRank, was developed to provide insight into the usefulness of each landmark in the online navigation algorithms described later in this research. The effectiveness of the NavRank metric was demonstrated by maintaining performance of the PnP algorithm when using significantly smaller databases, sub-sampled using NavRank.

While the algorithms derived in this chapter were heuristic in nature, the framework developed and derived here should provide useful for future research. The ability to analyze the database conditioned on the results of specific algorithms is relatively novel, and was shown to be useful in the case of deriving NavRank. This framework could be extended to provide insight into generation of visual-vocabulary models optimized for localization. This framework is also well suited for the study of temporal and platform specific effects on the usefulness of a landmark in navigation. The data used in this research was all from a single sensor, with little seasonal variation.

The schema developed here hints at the ability to dynamically choose the descriptor vector associated with the landmark at runtime. While discussed in the literature, there is no quantitative consensus on how to best update the appearance model (descriptor vector) of a landmark. Additionally, future work could also address the addition and pruning of landmarks from additional observations.

IV. Position Acquisition Algorithm

This chapter provides a derivation of an algorithm for image-based localization in the presence of large initial position uncertainty. A method for defining search extent given the initial position uncertainty from the INS mechanization is described. Finally, an approximation to the recursive Bayes filter in order to estimate a discrete location that is currently being observed by the airborne platform is derived.

4.1 Position Acquisition Algorithm Introduction

The large-scale image navigation algorithm is designed to estimate a coarse position of the aircraft given a prior position estimate with a large uncertainty. The output of this algorithm is then used to bootstrap a fine-tracking algorithm, which then provides accurate aircraft pose information to the core navigation algorithm.

The position acquisition algorithm outlined in this research provides an estimate of the ground footprint of an image acquired by the airborne platform. The initial position uncertainty from the presumably free-inertial navigation solution is used to define a search extent. The altitude of the camera and average terrain height are used to find the IFOV of a downward-looking camera located at the center of the search extent. The Spherical Mercator zoom level that creates tiles that bound this nominal downward-looking camera is used to create a grid over the search extent. A histogram filter is then used to compute a discrete probability distribution function that represents the likelihood that each of the grid cells generated the currently observed image. This observation likelihood is approximated using a novel direct feature-descriptor matching process, developed to address shortcomings in vision-vocabulary based techniques. The histogram filter is able to incorporate information from the strapdown-inertial mechanization to propagate the discrete probabilities between image updates.

This algorithm is evaluated using data from a flight test campaign conducted at Dugway proving ground. A reference database of keypoints was created from orthorectified satellite imagery. This Chapter evaluates the algorithm using 3 metrics:

percentage of observations where the true coarse pose was recovered, time to first fix, and the time required to generate the observation likelihood.

4.2 Wide Area Search Extent

The geographic area to be searched is defined by using the current position uncertainty from the aircraft navigation algorithm. The core aircraft navigator presents the position of the aircraft as a 3-dimensional Gaussian random variable:

$$\hat{\mathbf{p}}_0 = \mathcal{N}(\boldsymbol{\mu}_{\mathbf{p_0}}, \boldsymbol{\Sigma}_{\mathbf{p_0}}) \tag{81}$$

where $\boldsymbol{\mu}_{\boldsymbol{p_0}}$ is the mean and $\boldsymbol{\Sigma}_{\mathbf{p_0}}$ is the covariance matrix. The geographic area of the search extent was then defined as the area centered at $\boldsymbol{\mu}_{\boldsymbol{p_0}}$, with north and east ranges equal to three times the standard deviation in each direction. Once the geographic area of the search extent was established, the search area was discretized onto a grid whose cells roughly bound the ground footprint of the image observed by the airborne camera when pointing in the nadir direction.

A local-level coordinate system is instantiated with the origin located at the center of the search extent. The origin point is referenced as $\mathbf{t}_0^{ll}$. A Digital Elevation Model was used to find the vertical height of the terrain at the center of the extent, and this value is defined to be the vertical component of the origin of the local-level coordinate system.

Given the height of the aircraft above terrain from the aircraft core navigation algorithm, h_{cam}, the position of the notional aircraft camera in the new local-level frame is assumed to be directly above the center of the search extent: $\mathbf{t}_{\text{cam}}^{ll} = [0, 0, -h_{\text{cam}}]^T$. A camera looking toward the nadir direction will have its z-axis aligned with the down axis of the local level coordinate frame. As this is only a notional, approximate pose used to create the grid-size, the heading of the camera in this nominal pose is irrelevant, and the camera can be assumed to be axis-aligned with the local-level coordinate system, making the DCM between the two frames the identity matrix.

This relative geometry, along with the camera model is then used to determine the projected coordinates of the corners of the image in the local-level coordinate system. The back-projection equations given in Section 2.4.3 are used to calculate where a ray cast through each of the corners of the image intersects the north/east plane of the local-level coordinate system. The distance between the edges that join each successive projected images corner are calculated, denoting the longest length: l^{ll}_{proj}. To simplify the grid creation process, the maximum zoom level in Spherical Mercator whose tile edge size is still larger than l^{ll}_{proj} is used to define a grid over the geographic search extent. Each grid cell is modeled as a distinct location, for a total of n locations given a grid shape of $n = M \times N$.

To subsample the keypoint database, a maximum number of features that can be searched at any given image measurement, k, is defined. This upper bound is implementation specific, and depends on the computational capacity of the platform, along with time constraints for performing the estimation task. The algorithm derived in this dissertation was evaluated for k values: 125K, 250K, 500K, and 1M.

For each location in the model, the $j = k/n$ 'best' keypoint descriptors stored in the reference database are retrieved. Currently, the response value from the keypoint detector is used as the metric for 'best' features. In addition, the nominal GSD of an image defined at the center of the search extent, GSD_0 is calculated using the ground footprint. This value is used to exclude selecting keypoints from the reference database whose absolute size is smaller than GSD_0. The k loaded features forms the set $\mathcal{R}$.

4.3 Large-Scale Image Navigation Histogram Filter

This section outlines the formulation of the histogram filter used for the large scale search. The goal of the estimator is to determine the probability that the current image was generated from the observation of any of the grid cells within the defined search extent.

4.3.1 Location Model. This approach differs from that of FAB-MAP by embedding the appearance-space locations onto a metric 2D projection of the earth's surface. The global set of n locations is composed from each grid cell in the search extent, denoting this: $\mathcal{L} = \{L_1, \ldots, L_n\}$. The model for a given location L_i are the corresponding j feature descriptor vectors loaded in the search extent.

4.3.2 Measurement Model. At time t, an image from the airborne camera, I_t is received. The feature detector and descriptor extractor is used to generate the measurement vector, $\mathbf{z}_{t_k}$. This measurement vector is formed by retaining f keypoints from each image, ranked again by the response of the feature detector. The algorithm derived in this paper is evaluated for using $f =$ [1K, 5K, 10K, 20K] numbers of query features. The measurement vector $\mathbf{z}_{t_k}$ consists of the f keypoint descriptors sampled from image I_t. The set of all measurement vectors up to and including time t_{k-1} is denoted $\mathcal{Z}^{t_{k-1}}$ and the set of all measurements including $\mathbf{z}_{t_k}$ is denoted $\mathcal{Z}^{t_k}$. Similarly, the set of all control vectors from time t_0 to time t_k is denoted as $\mathcal{U}^{t_k}$.

4.3.3 Histogram Filter Formulation. The goal of the algorithm is to estimate grid location that is being imaged by the airborne platform at time t. As this algorithm is used to bootstrap other local-area navigation methods, the estimated location of the ground footprint of the image is more useful than the actual aircraft location. When the camera is pointed in the nadir direction, the ground footprint of the camera is directly underneath the aircraft, leading to the situation where the coarse aircraft position and image footprint are the same. For non-nadir pointing situations, the coarse aircraft position could be recovered using the IMU and the altitude of the aircraft.

A recursive Bayesian estimation strategy that maintains a probability over all $L_i \in \mathcal{L}$ is implemented as a histogram or grid filter, derived in [10]. At each time t, the probability that a given location L_i is currently being observed by the airborne camera is estimated by evaluating:

$$p(L_{i,t_k}|\mathcal{Z}^{t_k}) = \frac{p(\mathbf{z}_{t_k}|L_{i,t_k}, \mathbf{u}_{t_k}, \mathcal{Z}^{t_{k-1}})p(L_{i,t_k}|L_{i,t_{k-1}}, \mathbf{u}_{t_k}, \mathcal{Z}^{t_{k-1}})}{p(\mathbf{z}_{t_k}|\mathcal{Z}^{t_{k-1}})} \tag{82}$$

For each discrete location L_i a posterior probability distribution function $p(L_i|\mathcal{Z}^t)$ is evaluated. This represents the probability that the aircraft is currently observing the discrete location L_i, conditioned on all previous observations. The term $p(\mathbf{z}_{t_k}|L_i, \mathbf{u}_{t_k}, \mathcal{Z}^{t_{k-1}})$ is the observation likelihood, and approximates the probability that the measurement vector $\mathbf{z}_{t_k}$ was generated by location L_i, conditioned on the previous state. The filter allows for the integration of other navigation sensors through the motion model, and is represented by the prior term $p(L_{i,t_k}|L_{i,t_{k-1}}, \mathbf{u}_{t_k}, \mathcal{Z}^{t_{k-1}})$. The control vector $\mathbf{u}_{t_k}$ propagates the posterior probability from time t_{k-1} to time t_k using the aircraft INS and altimeter. Finally the equation is normalized using the $p(\mathbf{z}_{t_k}|\mathcal{Z}^{t_{k-1}})$ term. As this is a localization problem, the normalization is applied such that the sum of all $p(L_{i,t_k}|\mathcal{Z}^t)$ in $\mathcal{L}$ is equal to 1. A more detailed description of these terms and their derivation as applied to the image-localization algorithm is given in the following sections.

4.3.4 Prior Observation Term and Motion Model. The histogram filter formulation requires the use of a prior estimate for each location $p(L_{i,t_k}|\mathcal{Z}^{t_{k-1}})$. Initially at time t_0, each location maintains an equal probability.

During regular operation, the other sensors on board the aircraft (IMU, barometric altimeter, magnetic compass) are used to propagate the individual probabilities from the time of the last image update $t-1$ to the time of the current image t. The strapdown inertial navigation equations in [2] are used to calculate a delta-position vector $\Delta\mathbf{p}^{ll}_{\text{cam}}$ and covariance matrix $\mathbf{\Sigma_p}$. Using the projection algorithm described in Section 4.2, the center point of the camera is projected onto the XY plane of the nominal local-level plane, denoting the trajectory of the projection of the camera center point as $\Delta\mathbf{p}_{center}$.

Each location in the grid, $L_i \in \mathcal{X}$, has a corresponding 2D center point, $\mathbf{p}_i^{t_{k-1}}$. The motion model is applied to shift the grid by $\Delta\mathbf{p}_{center}^{ll}$ and carry forward the values of $p(L_{i,t_{k-1}}|\mathcal{Z}^{t_{k-1}})$ to the new grid centers $\mathbf{p}_i^t = \mathbf{p}_i^{t_{k-1}} + \Delta\mathbf{p}_{center}^{ll}$. An interpolation scheme is used to calculate the values of this pdf at the original grid center locations. During this interpolation, a Gaussian smoothing kernel with a region of support relative to $\mathbf{\Sigma_p}$ is used to account for uncertainty in the motion model. The result of this process is the resulting calculation of the probability that each of the grid cells $L_i \in \mathcal{L}$ are currently being observed by the aerial platform at time t_k, conditioned on the previous state, and inertial system delta-position vector between time t_{k-1} and t_k.

4.3.5 Observation Likelihood. In this section, an ad hoc approximation to the true observation likelihood term, $p(\mathbf{z}_{t_k}|L_{i,t_k}, \mathbf{u}_{t_k}, \mathcal{Z}^{t_{k-1}})$ is derived. The true likelihood is the probability that the current image is observed, given the prior probability of observing one of the grid locations, $L_i \in \mathcal{L}$. The described approach approximates this probability by creating a function of feature descriptor comparisons whose value is considered to be proportional to $p(\mathbf{z}_{t_k}|L_{i,t_k}, \mathbf{u}_{t_k}, \mathcal{Z}^{t_{k-1}})$. This approximation is generated through comparison of the f feature descriptors in the observation vector, $\mathbf{z}_{t_k}$, to the currently loaded subset of feature descriptors from the database, $\mathcal{R}$.

Each feature in the measurement vector $z_u \in \mathbf{z}_{t_k}$ is compared to every feature in the reference set, $r_v \in \mathcal{R}$ for the two features that are closest in appearance to z_u. A distance metric is used as a comparison between the observed and reference feature descriptor vectors, denoted d_{uv}. The distance metric chosen will depend on the choice of keypoint descriptor vectors. For example, SIFT can use the L2 distance metric, while binary feature descriptors may use a Hamming distance.

Using either linear search or one of the approximate k-nearest neighbor search algorithms described in [39], the feature descriptors in $\mathcal{R}$ with the smallest values for d_{uv} are indexed as $[uv_0, uv_1]$. Each feature $z_u \in \mathbf{z}_{t_k}$ is assigned a weight w_u which is the complement of the ratio of the distance of the two nearest neighbors:

$$w_{uv_0} = 1 - \frac{d_{uv_0}}{d_{uv_1}} \tag{83}$$

This weight function is used as an approximation to the probability that the reference feature r_{v_0} matches the observed feature $z_u \in \mathbf{z}_{t_k}$. The grid cell location of reference feature $L_{r_{v_0}}$ is used to bin the match weight w_{uv_0} into the corresponding set of weights. This results in a collection of sets of weights for each location in the grid, $W_i \in \mathcal{W}$. Each set of weights W_i that corresponds to location L_i consists of either the empty set, or weights: $W_i = \{w_0, \ldots, w_b\}$, where $b < f$. An illustration of the approximate observation likelihood function is shown in Figure 17. To approximate the probability that the measurement vector $\mathbf{z}_{t_k}$ was generated by observing location L_i, the weights binned in the corresponding set, W_i are accumulated:

$$p(\mathbf{z}_{t_k} | L_{i,t_k}, \mathbf{u}_{t_k}, \mathcal{Z}^{t_{k-1}}) \approx \sum_{l=0}^{b} w_l \tag{84}$$

For locations whose corresponding weight-set is empty, a small, non-zero probability is used to account for process noise and maintain numerical stability. This approach is admittedly an approximation to the actual observation likelihood. Therefore, no formal derivation showing optimality of the algorithm is given, although the algorithm design still follows the concept of the Bayesian approach. This approach was taken as it is not at all clear how one would determine the true probabilities of feature matches, given only their descriptors.

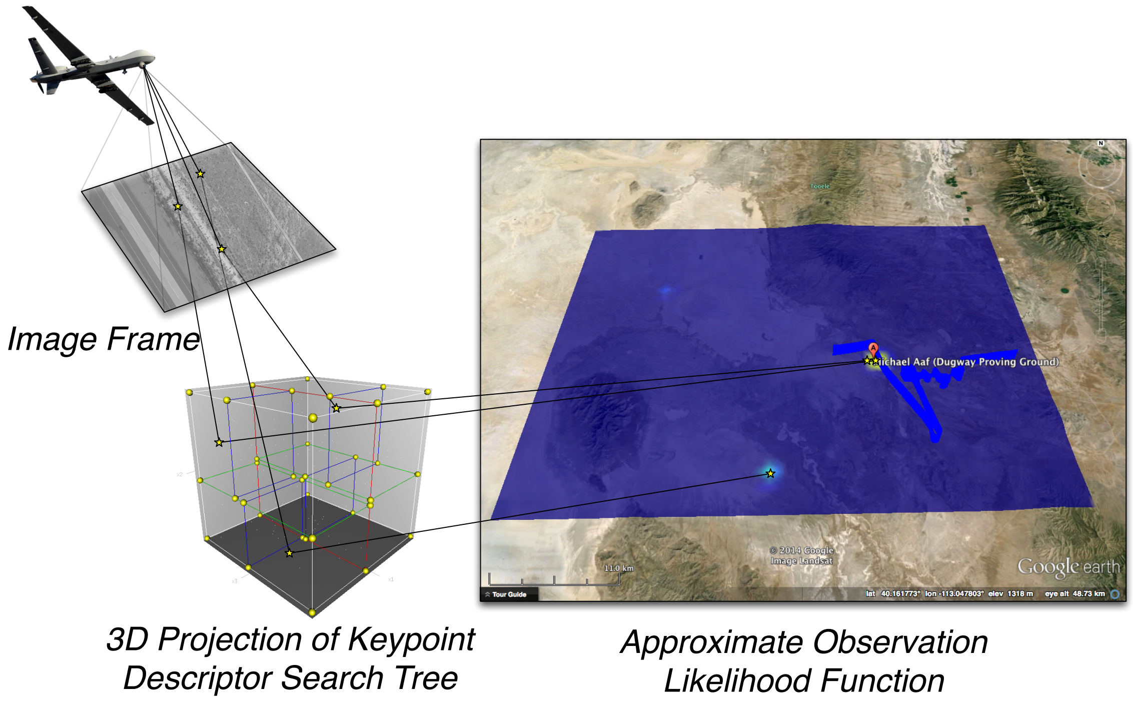


Image Frame
3D Projection of Keypoint Descriptor Search Tree
Approximate Observation Likelihood Function
Michael Aaf (Dugway Proving Ground)
Google earth

Figure 17: Illustration of the approximate observation likelihood function used in the wide-area search algorithm.

4.3.6 Iteration of Histogram Filter. With each observation vector $\mathbf{z}_{t_k}$ the motion model is used to propagate $p(L_{i,t_{k-1}}|\mathcal{Z}^{t_{k-1}})$ to the time of the image, t_k. The update step is then applied by evaluating the approximation to the observation likelihood. Finally, the resulting probabilities are normalized by the magnitude of the probability vector, $p(\mathcal{L})$. The resulting distribution is the posterior probability that the airborne camera is currently observing any of the grid locations.

While not derived in this research, it is theorized that there is some unknown minimum number of keypoint descriptors, j_0, that must be loaded for a given search extent in order to provide useful observations. This minimum number would be the bound for the scalability of a direct feature matching approach. While no theoretical derivation for this value is given here, some empirical results are shown in the following section using various values of j.

4.4 Implementation and Flight Test Campaign

In order to test the algorithm, a series of flight test campaigns were performed at Dugway Proving Grounds. The sensors used include a Prosilica GE2040 grayscale machine-vision camera, with a resolution of 2048x2048. The camera used a high-quality Canon 50mm EF lens. To maintain calibration, the focus adjustments were set to focus at infinity and glued together. Altitude was provided using a barometric altimeter. Additionally, a Novatel SPAN system, consisting of a Novatel OEM-V GPS receiver and a tactical-grade IMU was used. The GPS measurements were used only for system initialization and to provide truth data.

The majority of the time spent observing the database area was flown at an altitude of roughly 1700m above ground level (AGL). This provided a ground footprint of about 800x800m when the camera was looking downwards. The discrete feature grid was set to the Spherical Mercator Level 15 zoom, which provides 930x930m tiles around Dugway. For the wide-area search, only a nominal camera calibration is needed. However, ground points across Dugway proving grounds were surveyed and used an offline calibration routine in order to estimate the extrinsic camera to IMU lever-arm and relative rotation parameters, and the intrinsic pinhole-camera model parameters. A description of this process is provided in Chapter 5.3.

4.4.1 Software Implementation. The wide-area search algorithm was implemented in Python, using both the OpenCV [42] and scikit-image [43] libraries for image processing. The sensor drivers and SIFT feature extraction were written in C++, and interprocess communication was accomplished through the use of the Robotic Operating System (ROS) [44]. The GPU-accelerated SIFT implementation found in [45] was used for SIFT keypoint detection and subsequent HoG descriptor extraction. Soft real-time SIFT descriptor extraction was demonstrated on the relatively low-powered Intel HD 4000 GPU in the 2012 Apple Mac Mini. Observation likelihood generation used the Fast Library for Approximate Nearest- Neighbors (FLANN) [39]. The SIFT feature database was implemented using the PyTables [33] library. The features were stored by their grid-cell unique index, allowing for fast retrieval and queries.

4.5 Results

To quantify the results of the wide area search, three metrics are examined using a single, representative flight from the Dugway flight test campaign. The first metric is Time to First Fix (TFF), defined as the number of images that were observed by the aerial platform in the database area before the peak of the posterior of the wide-area search converged to the tile that generated the current image. The second metric is correct detection rate, and measures the percentage of times that the peak of the posterior distribution overlaps with the currently observed tile. For instances where the camera footprint overlaps multiple discrete grid locations, overlap of the peak of the posterior with any of those tiles are identified as a correct detection. The last metric is the average amount of time required to generate the observation likelihood function. The nearest-neighbor search algorithm has an $O(jk \log k)$ complexity (j being the number of features in the airborne image after windowing, and k being the total number of features loaded from the database). These metrics were evaluated for varying numbers of j and k.

4.5.1 Wide Area Search Results. These metrics are currently computed for a single, typical flight of the 10 conducted during the flight test campaign. Figure 18 shows a black line representing the projection of the center point of the image in Spherical Mercator space. This projection is overlaid on the sum of the posterior of the wide area search for the $j = 10^3, k = 1 \times 10^6$ case. This flight is composed of 1100 observations of the database area. At 0.5 Hz, this is a period of roughly 37 minutes. Figure 19 shows the same aggregation of the posterior estimate for varying values of j and k. Again no detailed camera calibration is used, and the full extent of k features is used for the sole observation likelihood. The INS mechanization is used to provide inputs for the motion model between observations. These results are detailed for the

SIFT approximate observation likelihood generation function, with some comparative results from a SURF-based observation likelihood function.

The resulting search correct detection rates for SIFT keypoints and descriptors are given in Table 2, for varying parameters of j and k. As the number of features searched (k) is increased, correct-detection rates increase significantly. An interesting point is that when increasing the number of query features (j) used to construct the observation likelihood function, the algorithm does not generate statistically significantly better performance. Also, as the number of database features grows toward $k = 20 \times 10^3$, there was a significant decrease in correct detection rate. This result, while counter-intuitive, is potentially explained in [39]. As the number of reference features k increases, the probability that a feature $z_u \in \mathbf{z}_{t_k}$ returns a false nearest-neighbor also increases. Therefore, implementations of this algorithm should attempt to bound the values of j and k to the values that provide sufficient performance for a particular application.

The f keypoints from the query image are selected first by scale-windowing, and then selected in order of detector response. For this dataset, the average number of features extracted from the observed images was $48,000$, a max of $165,000$, with 14 of 1349 images having no features. As the number of features per image, f, is increased, performance was stable until the $f = 5^3$, and then decreased. This indicates that most of the informative features from the image are also identifiable by response. Furthermore, these features represent a small percentage of the total number of extracted features.

Table 3 shows the time needed to generate an approximate observation likelihood using SIFT keypoints and descriptors. This process scales with $O(jk \log(k))$, therefore we are looking for the smallest values of j and k that provide our desired correct detection rates. These results were computed using the Python language FLANN bindings on a Quad-Core Intel 2.7 GHz Intel Core i7 (Early 2013 Apple Retina MacBook Pro). Notice that the highest correct detection rate, 70.78%, occurred with

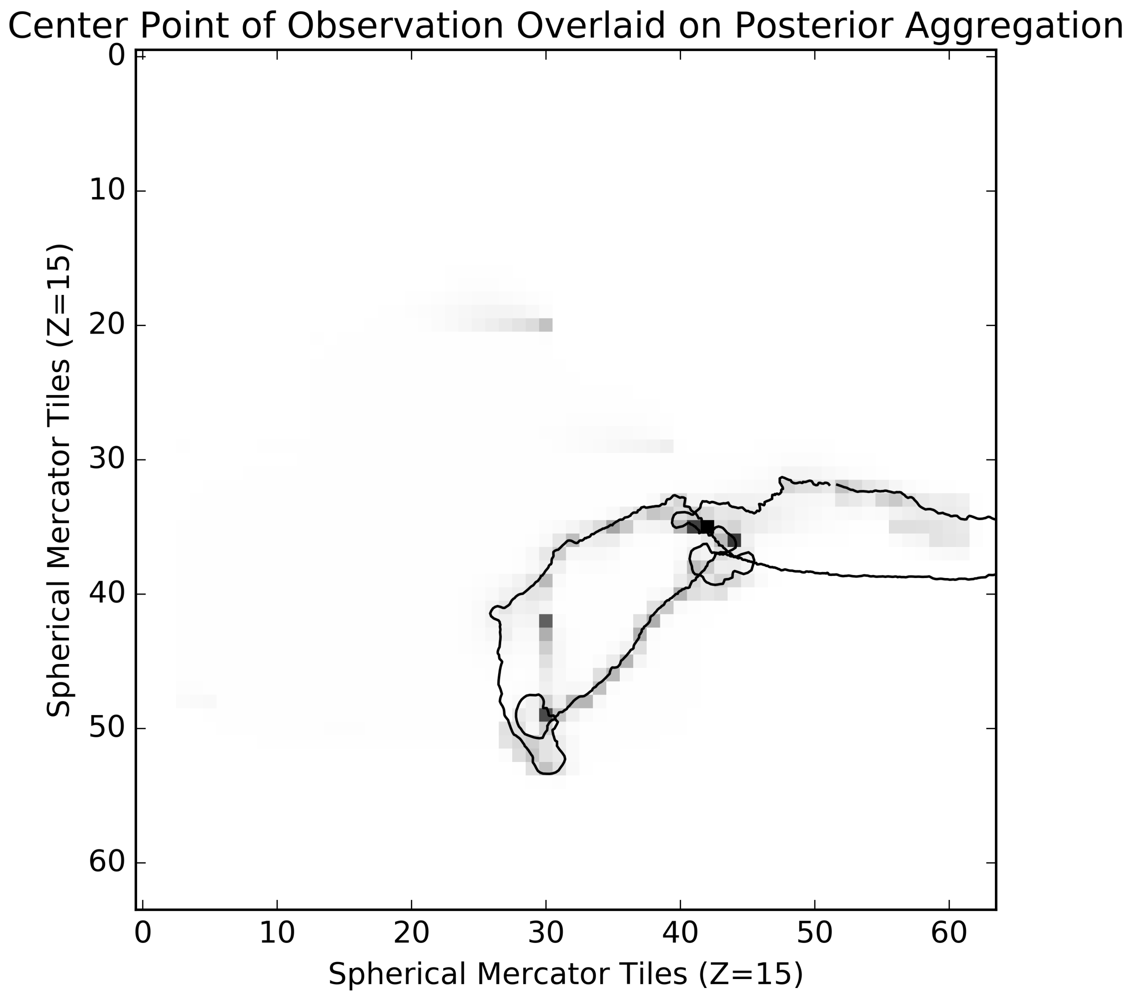


Figure 18: Sum of posterior estimate for a single flight using 500K SIFT reference landmarks and 5k query keypoints per image. Ground track of the center of the image is represented by the black line. Units are in Spherical Mercator Tiles at Zoom 15 (roughly 930m).

Table 2: Wide Area Search Correct Detection Percentages for SIFT

k=Reference Features	j = Number of Observed Image Features			
	1K	5K	10K	20K
125K	51.6%	44.6%	36.3%	16.7%
250K	53.7%	54.9%	40.7%	17.7%
500k	64.0%	58.8%	55.6%	47.3%
1M	66.8%	70.8%	69.4%	64.0%

Table 3: Wide Area Search Observation Likelihood Generation Time for SIFT in Seconds

k=Reference Features	j = Number of Observed Image Features			
	1K	5K	10K	20K
125K	0.0075s	0.0253s	0.0452s	0.0792s
250K	0.0101s	0.0361s	0.0601s	0.1035s
500K	0.0426s	0.1829s	0.3364s	0.6996s
1M	0.1899s	0.8372s	1.5814s	2.8665s

Table 4: Wide Area Search Time To First Fix (TTFF) for SIFT in Seconds

k=Reference Features	j = Number of Observed Image Features			
	1K	5K	10K	20K
125K	306s	436s	436s	436s
250K	304s	304s	436s	456s
500K	306s	304s	304s	442s
1M	306s	292s	292s	292s

j =5K and k =1M for an observation likelihood generation time of 0.84s. With a framerate of 0.5Hz, real-time performance was maintained for the wide area search.

Finally, the Time to First Fix for SIFT keypoints and descriptors are shown in Table 4. This metric is time difference between the time of first entering the database region, and the time when the peak of the search posterior coincided with the true camera footprint. For values of j and k where correct detection was above 50%, the TTFF was roughly equal at about 5 minutes. We note that the ingress/egress point of the database area (far right of Figure 18) is relatively low in feature density. Further analysis of the data is needed to study the effect of trajectory (e.g., order of traversal of the database) on algorithm performance.

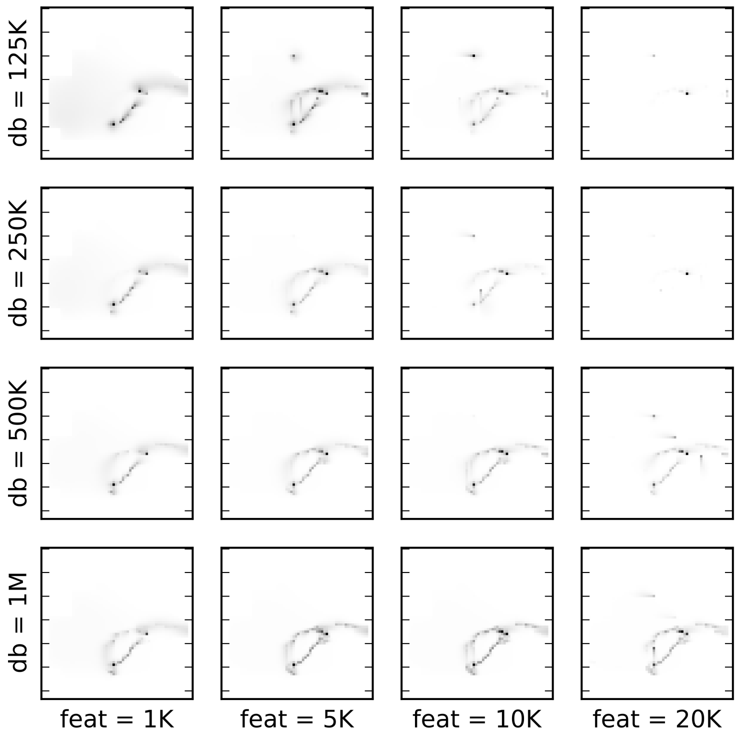


Figure 19: Sum of Posteriors of Wide Area Search Algorithm for Varying Size of Query Features and Reference Features Searched

This algorithm was also evaluated using the SURF keypoint detector and descriptor described in Chapter III. The resulting aggregation (sum) of the posterior for the 1K query keypoint and 1M reference landmark case is shown in Figure 20. The metrics for this case show a much longer time to first fix, at 686.0 seconds. While, the detection percentage for this case, 54.34%, was significantly lower than the same parameter case for SIFT keypoints, correct detection performance after the first fix significantly increased to 72.86%. SURF observation likelihood generation was significantly faster in all cases, and for the studied case showed a factor of 5 improvement ($0.0390s$).

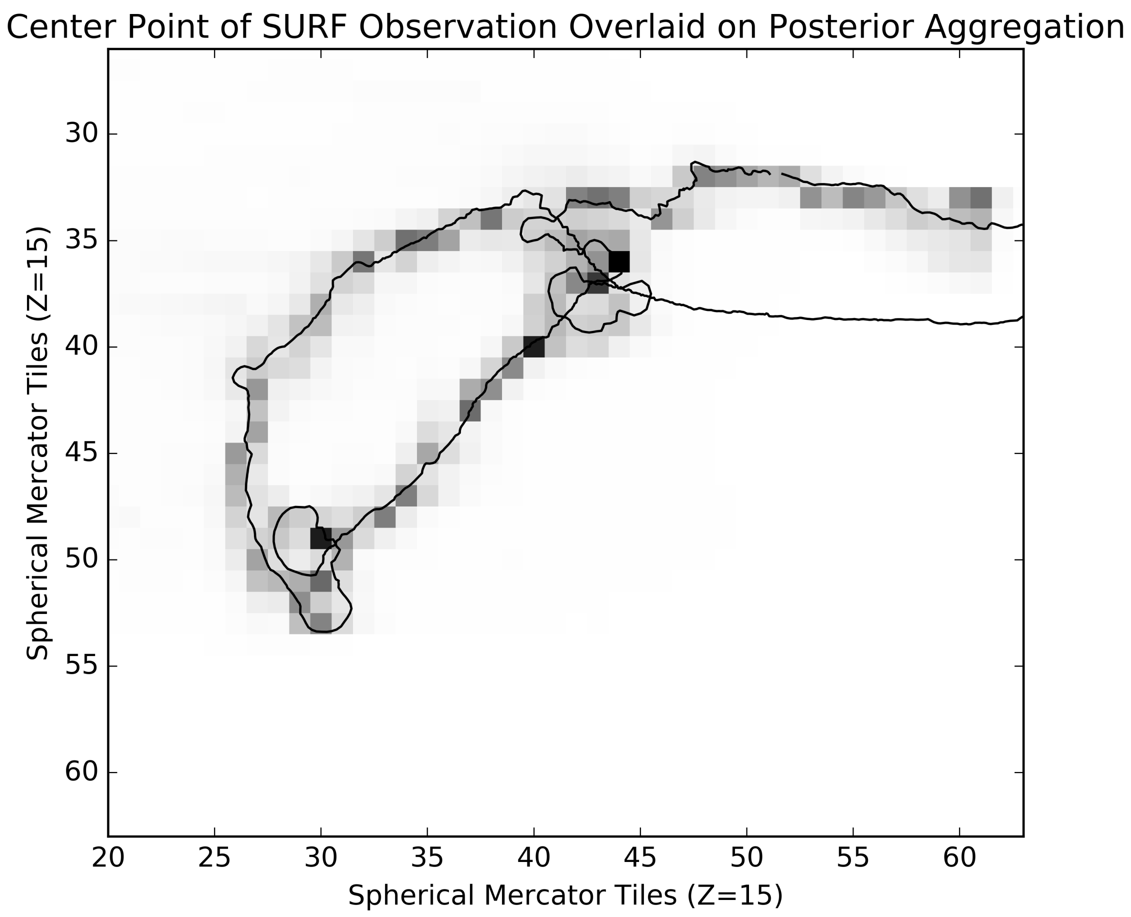


Figure 20: Sum of posterior estimate for a single flight using 1M reference SURF landmarks, and 1k query keypoints per image. Ground track of the center of the image is represented by the black line. Units are in Spherical Mercator Tiles at Zoom 15 (roughly 930m).

4.6 Chapter Summary

In this Chapter, a wide-area search algorithm capable of reducing a large initial uncertainty (45km by 55km) resulting from INS position drift, to that of the footprint of the airborne camera (900m x 900m), was derived and implemented. The database pipeline outlined in Chapter III was used to provide a subset of the 'best' keypoints for wide-area search. A novel approximation to the observation likelihood function was derived and implemented by using the projection of the results of an appearance-based nearest neighbor feature search onto a discrete location grid.

In addition, a histogram filter using the approximate observation likelihood function was derived. This filter used inputs from the INS to propagate the prior distribution through time. This algorithm achieved a 70% correct detection rate using flight test data. This algorithm was evaluated using two different keypoint detection and descriptions methods, SIFT and SURF.

Several areas of future research have been identified. While GNSS-based navigation systems have a relatively understood error model, feature-based navigation performance depends greatly on the database being used for comparison. Additional quantitative analysis is needed to understand the effect of database quality on this approach. Additionally, the trajectory in which the database is traversed is anticipated to have an effect on performance. In addition, this research suggests that the offline learning steps shown here may be useful in creating a vision-vocabulary that could produce better results for CBIR-based approaches to airborne navigation.

V. Fine Tracking Algorithm

Using a set of georeferenced keypoints to provide real-time navigation solutions for airborne platforms is an area of active research. This Chapter outlines contributions to the state-of-the art for determining aircraft position using a database of landmarks in a flight environment.

5.1 Overview of the Fine Tracking Algorithm

This chapter details the implementation of the Perspective-n-Point (PnP) algorithm, optimized for use in a flight environment. This algorithm is referred to as a *fine-tracking* algorithm, as the output of the algorithm is able to estimate the pose of the airborne camera to GPS-level accuracy (less than 20m three-dimensional RSS error). An overview of the algorithm is given in Figure 21.

The algorithm starts with using the coarse pose provided by the position acquisition algorithm described in Chapter IV. In the implementation of the algorithm, this coarse pose is given as the most likely Spherical Mercator tile that bounds the IFOV of the currently observed image. A query is made into the landmark database to retrieve a set of landmarks within the most likely Spherical Mercator tile. In addition, the aircraft altitude and pinhole-camera model are used to compute the effective absolute scale range of the image, and that range is used to exclude landmarks from being retrieved from the database that are too small to be observable.

After retrieving a set of reference landmarks, a nearest-neighbor matching algorithm is used to find the two closest landmarks for each keypoint generated by the observed image. A ratio test is used to discriminate against weak matches in appearance space. The resulting 2D/3D correspondences are then used by the Perspective-n-Point algorithm to estimate camera pose. This research also shows that using the *a-priori* estimate of camera attitude from the INS reduces the error of the resulting position estimate, on average.

The use of the PnP algorithm for pose estimation is well established [1], [46]. This research claims several novel modifications to optimize the use of PnP for the

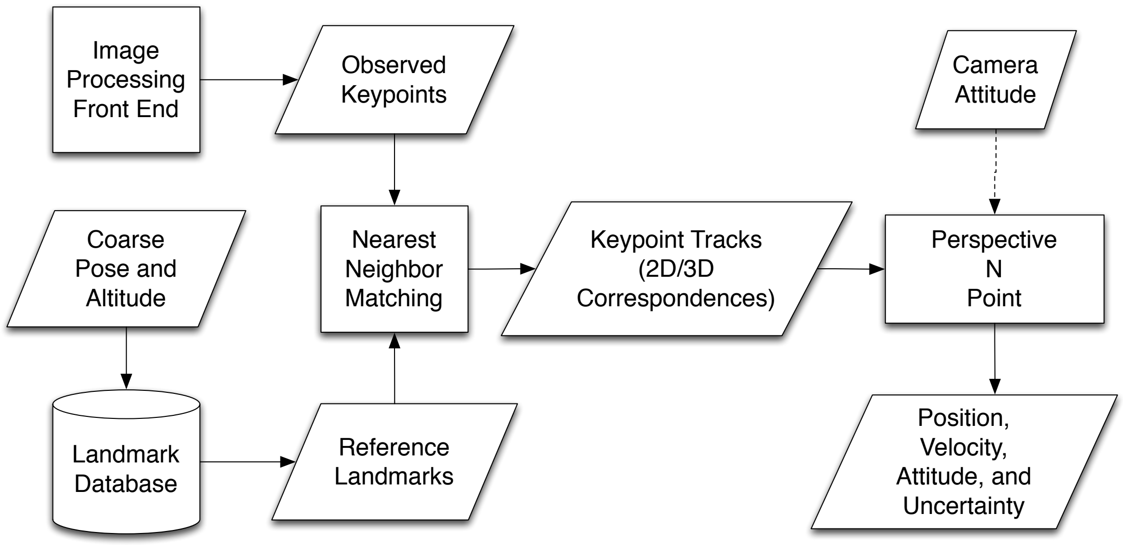


Image Processing Front End
Observed Keypoints
Camera Attitude
Coarse Pose and Altitude
Nearest Neighbor Matching
Keypoint Tracks (2D/3D Correspondences)
Perspective N Point
Landmark Database
Reference Landmarks
Position, Velocity, Attitude, and Uncertainty

Figure 21: Overview of the proposed fine-tracking algorithm.

flight environment. First, the use of coarse position along with the *a-priori* estimate of camera pose from the INS to load a subset of landmarks is a novel optimization. Second, most implementations of PnP attempt to estimate the full six degree of freedom estimate of camera pose. This research shows that using the attitude estimate from the INS, and instead only estimating the three degree-of-freedom translation estimate provides a significant accuracy benefit, specifically in situations where the PnP solution may be poorly-conditioned due to keypoint/landmark geometry. Third, this research proposes the re-use of the 2D/3D correspondences generated as an artifact of the PnP pose estimation process in camera-calibration and landmark-database refinement algorithms.

5.2 Perspective-n-Point

Perspective-n-Point is a class of algorithms that uses n correspondences between detected 2D keypoints in an image, and their matched 3D landmarks in order to estimate camera pose. There are closed-form solutions for pose estimation for a specific number of points [1], [47]. However, PnP is generally implemented in some type of estimation strategy to account for noise in the observations, as well as using information from an overdetermined solution to increase pose estimate accuracy.

PnP estimates the rotation of the camera with respect to the world frame at time t_k, $\mathbf{R}^w_{c,t_k}$, and the translation of the camera in the world frame, $\mathbf{p}^w_{t_k}$, given correspondences between world points $\mathbf{X}^w$ and measurements $\mathbf{z}_{t_k}$ of points in the image frame $\mathbf{x}^{img}_{t_k}$. Each 2D/3D correspondence provides independent equations, therefore to solve for the six degrees of freedom for camera pose, at least 3 such correspondences are needed. For robustness, more points are used to check for consistency, as well as increase accuracy. The projection of the j^{th} point $\mathbf{X}^w_j$ onto the image plane using the process in Equation (46), is represented by a function $h(\mathbf{K}, \mathbf{R}^w_{c,t_k}, \mathbf{X}^w_j, \mathbf{p}^w_{t_k})$. Equation (46) is applied to generate a 3D location in the image frame:

$$\begin{bmatrix} x^{img}_j \\ y^{img}_j \\ z^{img}_j \end{bmatrix} = \mathbf{K}\left[\mathbf{R}^c_{w,t_k}\mathbf{X}^w_j - \mathbf{R}^c_{w,t_k}\mathbf{p}^w_{t_k}\right] \tag{85}$$

where the measured keypoint locations are 2D coordinates on the image plane. The function h normalizes the result of Equation (85) by the z coordinate to provide the 2D projection of the landmark onto the image frame given the current estimates of camera rotation and translation:

$$h(\mathbf{K}, \mathbf{R}^w_{c,t_k}, \mathbf{X}^w_j, \mathbf{p}^w_{t_k}) = \left[\frac{x^{img}_j}{z^{img}_j}, \frac{y^{img}_j}{z^{img}_j}\right]^T \tag{86}$$

Using this projection function, PnP employs a non-linear optimization strategy minimizing a cost function, $F_j(\mathbf{R}^w_{c,t_k}, \mathbf{p}^w_{t_k})$, defined as the difference between the measured keypoint location in the image $\mathbf{z}_{j,t_k}$ and the projected features from Equation (86). An illustration of a simplified PnP problem is shown in Figure 22. This cost function is given as:

$$F_j(\mathbf{R}^w_{c,t_k}, \mathbf{p}^w_{t_k}) = \| \, \mathbf{z}_{j,t_k} - h(\mathbf{K}, \mathbf{R}^w_{c,t_k}, \mathbf{X}^w_j, \mathbf{p}^w_{t_k}) \, \|^2_{\boldsymbol{\Sigma}_{j,i,t_k}} \tag{87}$$

where $\|\mathbf{q}\|_{\boldsymbol{\Sigma}}$ is the Mahalanobis Norm of $\mathbf{q}$, and the covariance matrix of the 3D to 2D projection operation of the j^{th} point at time t_k is represented by $\boldsymbol{\Sigma}_{j,i,t_k}$. The Mahalanobis Norm can be transformed to the L2-norm by providing the matrix square root of $\boldsymbol{\Sigma}$ as $\boldsymbol{\Sigma}^{-1/2}$ and the following relationship:

$$\|\mathbf{q}\|^2_{\boldsymbol{\Sigma}} = \mathbf{q}^T \boldsymbol{\Sigma}^{-1} \mathbf{q} \tag{88}$$

In order to incorporate uncertainty of the 3D position of the j^{th} landmark, the state vector, or values being estimated at time t_k are augmented to include an estimate of the 3D location of the landmark. Augmenting the projection function to include the location of the world point, $\mathbf{X}^w_j = \left[x^w_j, y^w_j, z^w_j\right]^T$, changes Equation (86) to:

$$h(\mathbf{K}, \mathbf{R}^w_{c,t_k}, \mathbf{X}^w_j, \mathbf{p}^w_{t_k}) = \left[\frac{x^{img}_j}{z^{img}_j}, \frac{y^{img}_j}{z^{img}_j}, x^w_j, y^w_j, z^w_j\right]^T \tag{89}$$

The augmented measurement vector for the j^{th} landmark is then given as:

$$\mathbf{z}_{j,t_k} = \left[\, \hat{\mathbf{x}}^{img}_j, \hat{\mathbf{X}}^W_j \,\right]^T \tag{90}$$

where $\hat{\mathbf{x}}^{img}_j$ is the measured position of the keypoint from the image processing front end, and $\hat{\mathbf{X}}^W_j$ is the measured landmark position in the world coordinate system, given by the reference landmark database. The covariance matrix is also augmented with the uncertainty information about the landmark position included in the reference database, $\boldsymbol{\Sigma}_{j,w}$, to create a 5×5 covariance matrix:

$$\boldsymbol{\Sigma}_{j,t_k} = \begin{bmatrix} \boldsymbol{\Sigma}_{j,i,t_k} & 0 \\ 0 & \boldsymbol{\Sigma}_{j,w} \end{bmatrix} \tag{91}$$

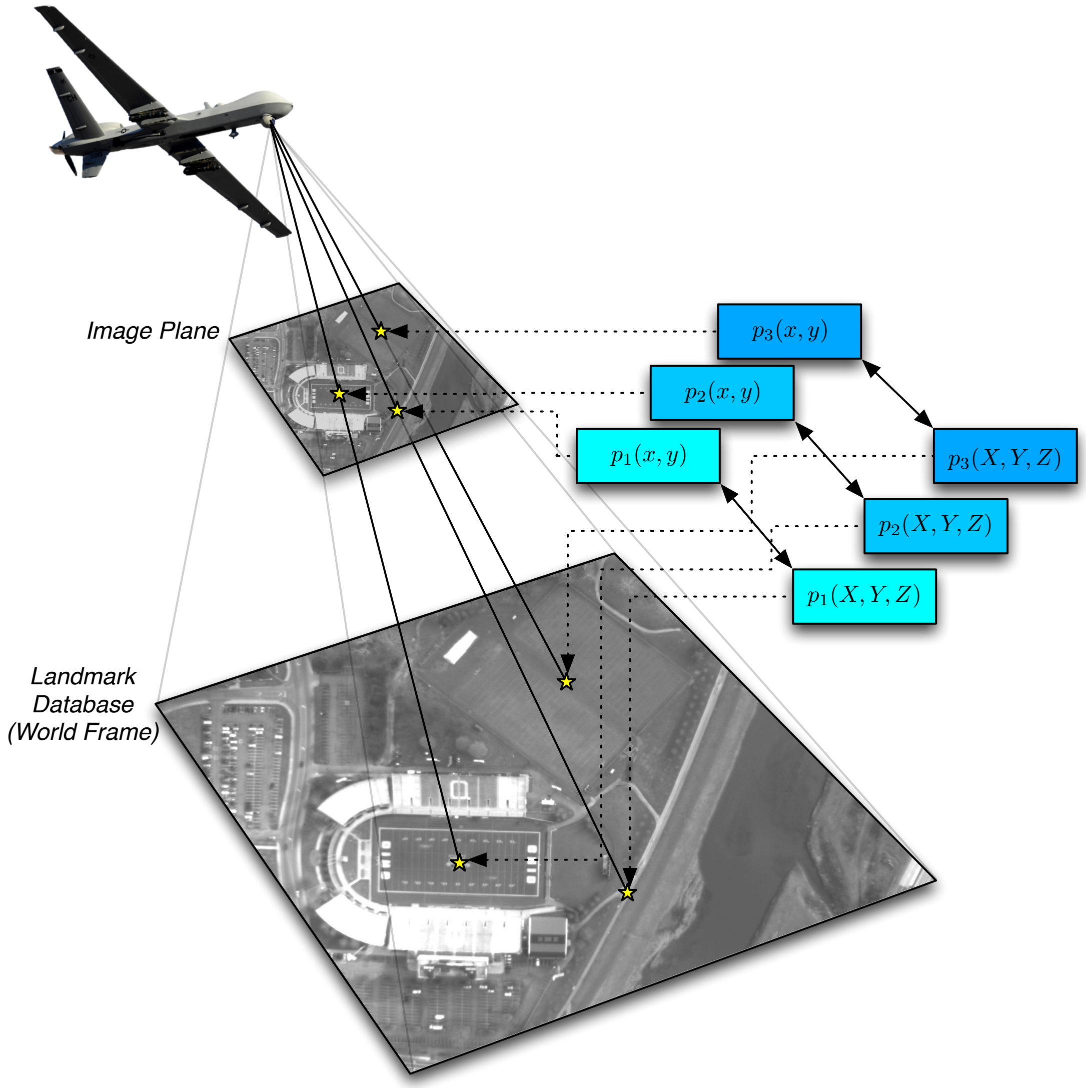


Figure 22: Illustration of the Perspective-n-Point algorithm being employed in a flight environment.

The augmented measurement vectors and projection functions are then used as replacements in Equation (87). For the n matched 2D/3D correspondences in any given image, the total cost function is the squared L2-norm of all the individual residuals:

$$\|\mathbf{F}(\mathbf{R}^w_{c,t_k}, \mathbf{p}^w_{t_k}, \mathbf{X}^w)\|_2^2 = \| \begin{bmatrix} F_1(\mathbf{R}^w_{c,t_k}, \mathbf{p}^w_{t_k}, \mathbf{X}^w_1) \\ \vdots \\ F_n(\mathbf{R}^w_{c,t_k}, \mathbf{p}^w_{t_k} \mathbf{X}^w_n) \end{bmatrix} \|_2^2 \tag{92}$$

The PnP problem is formulated as an estimation problem, such that the estimated values of camera rotation, translation, and world point locations minimize the squared L2-norm of the projection error:

$$\hat{\mathbf{x}}_{t_k} = \arg\min_{\mathbf{x}_{t_k}} \|\mathbf{F}(\mathbf{x}_{t_k})\|_2^2 \tag{93}$$

where the augmented state vector, $\mathbf{x}_{t_k}$ consists of the parameters that define the camera translation, rotation, and 3D locations of all the landmarks used at time t_k. The problem defined in Equation (93) is non-linear (affine), and this research uses an iterative, non-linear solver to compute the parameter vector that minimizes the cost function in Equation (92). Specifically, this research implements the Levenberg-Marquardt non-linear solver [48], [49]. The implementation of the PnP algorithm within the context of the Levenberg-Marquardt optimization scheme is provided in Appendix A, distributed separately from this document. The appendix can be obtained by request from the author, or the Air Force Institute of Technology, Autonomy and Navigation (ANT) Center.

5.2.1 Outlier Detection. The PnP model can be used in conjunction with the Random Sample Consensus (RANSAC) method to detect outliers in the set of matched 2D/3D correspondences [50]. RANSAC uses a random minimal subset of the 2D/3D correspondences to estimate a pose estimate of the camera. The initial pose model is then used to project the 3D points into the image plane, and points whose projected values are close to the measured values within some threshold are counted as inliers, with the rest as outliers. This process can then be rerun multiple times, iteratively refining the model. Thresholds can be used to provide some confidence that RANSAC returns the correct pose estimate of the camera, and correctly identifies between inliers and outliers.

5.3 Camera Calibration

In order for the PnP pose estimate to be accurate, the parameters of the camera matrix $\mathbf{K}$, along with the rotation and translation of the camera about the navigation system computation (body) frame, $\mathbf{R}^b_{\text{cam}}, \mathbf{t}^b_{\text{cam}}$ must be estimated. This process is known as camera calibration. Estimation of the parameters of the pinhole camera model is known as *intrinsic calibration*, where the estimation of the camera pose with respect to the body frame is known as *extrinsic calibration* [42].

The intrinsic parameters of a camera are typically calibrated using a multi-view formulation of the PnP problem, or a full bundle adjustment problem [42]. For many scenarios, calibration is accomplished using a calibration target, a grid of uniformly spaced geometric primitives which form a uniquely identifiable world coordinate system. An example of a checkerboard calibration target is shown in Figure 23. Images of the calibration target are generated from multiple camera in-plane rotations, translations, and affine viewpoints. The corners of the checkerboard are uniquely identified in each image, and the 3D / 2D correspondences are recorded.

The calibration problem is formulated as another non-linear least-squares optimization problem, where again the goal is the minimization of image-plane reprojection error, and the detailed implementation is provided in Appendix A.

5.3.1 Manual Camera Calibration. In the camera calibration process, the platform navigation system is used in an environment where GPS measurements are available to provide accurate estimates of aircraft position and rotation with respect to

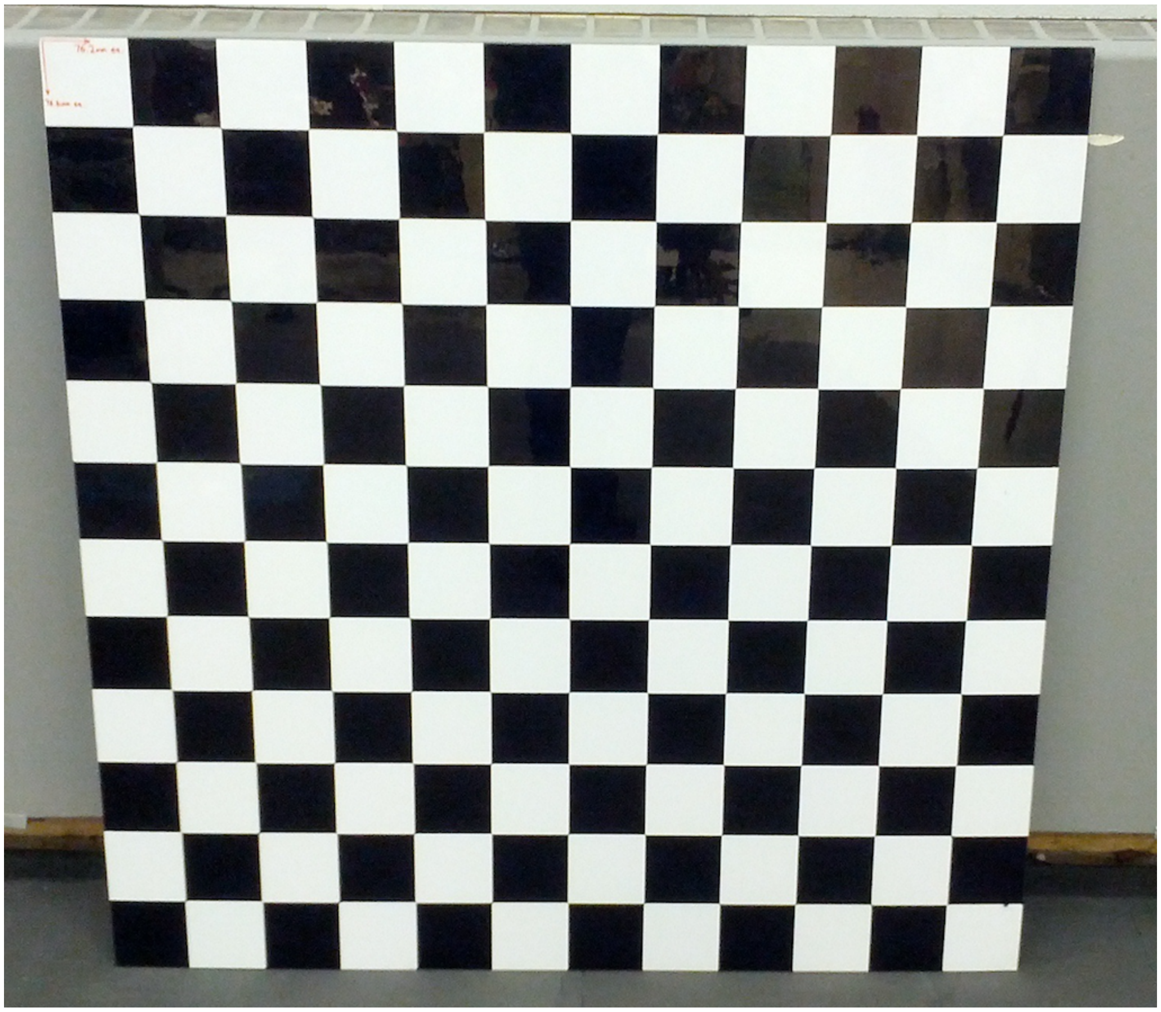


Figure 23: Example of a checkerboard camera calibration target.

a local-level world frame. When performing a manual camera calibration, world-points that can easily be identified in the platform imagery must be surveyed. Typically, survey-grade GPS equipment is used to estimate the WGS-84 location of intersection of runway markings, road markings, calibration targets, or opportunistic geometric primitives around a base of operations. An example of survey calibration points performed in the area where the experimental results of this research were generated is shown in Figure 24.

To generate the 2D/3D correspondences used in the optimization problem, an analyst will manually identify and record the pixel locations of each survey point that is observed in a given image. These manually generated correspondences are then used in the optimization routine to estimate the intrinsic, extrinsic, and bias parameters of the calibration state vector. While this process can provide accurate estimates of camera parameters, military users may not have the ability to perform accurate surveys of areas where the aircraft will be operating, nor wish to incur the analyst cost for manually identifying world points in calibration imagery.

5.3.2 Automatic Camera Calibration. This research claims that generating the 2D/3D correspondences for camera calibration by using artifacts from the PnP process provides a suitable set of calibration parameters that can then be used for later flights, without the manual intervention of either survey teams or image analysts.

Starting with a nominal estimate of camera calibration parameters either generated by a manual process, or by using the specifications of the imaging sensors and nominal mounting, the PnP pose estimation process is computed for each image in a calibration sequence. Assuming that each image in the calibration sequence also has accurate estimates of aircraft pose from the aircraft navigation system, the residual error between the PnP estimate is compared to estimate of aircraft translation from the truth system. If this value is within some predefined threshold (depending on the accuracy of the initial camera calibration parameters), the inlier 2D/3D correspondences are stored for use in the subsequent batch camera calibration process.

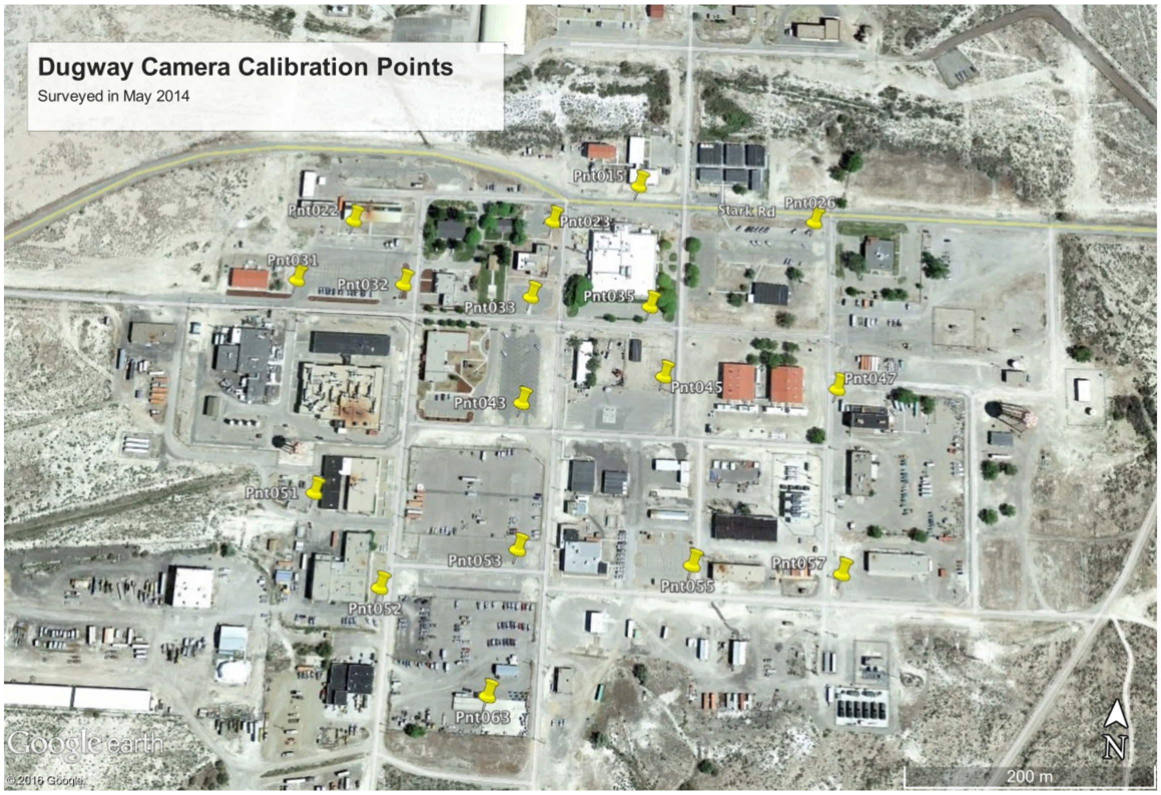


Figure 24: Manually surveyed camera calibration world-points around Dugway Proving Grounds. Image provided by Google.

This process allows for significantly more 2D/3D correspondences to be used in the calibration routine, from more varied parts of the aircraft trajectory. In addition, as the landmark database is providing the world-coordinates for the 2D/3D correspondences, the estimated bias from the calibration routine can then be re-incorporated into the landmark database to improve the performance of non-calibration flights, as detailed in Chapter III. Results of the automatic calibration routine using data from the flight experiments performed during this research are shown later in this chapter.

5.4 Landmark Retrieval and Matching

This section will outline the methods used to retrieve a set of landmarks from the reference database, given a coarse estimate of the camera footprint, camera altitude, and the camera model.

The position of the aircraft in the world frame $\mathbf{p}^w_{t_k}$ is established, either through the combination of the coarse position from the acquisition algorithm and an altitude measurement, or the *a-priori* estimate of position from the aircraft navigation system. Given an estimate of the aircraft camera frame to world frame rotation matrix from the aircraft INS, $\mathbf{R}^w_{c,t_k}$, the world-coordinates of a ray projected through the center of the image are calculated using the forward projection method described in Equations (52)-(54).

The estimated location of the forward-projected center point of the image, $\hat{\mathbf{c}}^w$, is then used as a query into the landmark database to retrieve k reference landmarks from a local area about $\hat{\mathbf{c}}^w$. In the tracking/PnP phase, k is a much smaller value than the wide area search. The results presented in this research used a fixed value of $k = 10\text{K}$ to generate PnP estimates of aircraft position. For this research, the k landmarks are retrieved using the heuristic defined in Chapter III for ‘best’ landmarks, along with some Euclidean distance threshold around $\hat{\mathbf{c}}^w$. The results presented in this Chapter used the Spherical Mercator tile at Zoom 15 which contains $\hat{\mathbf{c}}^w$ as a bounding box to obtain the best k landmarks.

For each image in the sequence, j query-keypoints are used to compare against the k landmarks loaded from the database. The results presented in this research use a fixed value of $j = 5\text{K}$. To match the j query keypoints against the k landmarks, a brute-force matching strategy is applied to eliminate any randomness in the results that may be incurred when implementing an approximate nearest neighbor strategy. The brute-force matching strategy computes a distance between each query keypoint descriptor vector and the descriptor vector associated with every reference landmark. A ratio test is used to compare the distances between the two closet landmark descriptor vectors, and if the smallest distance is at least τ times smaller than the second closest, the query keypoint and landmark pair is kept as a candidate 2D/3D correspondence.

For each set of resulting 2D/3D correspondences, the outlier detection algorithm described in Section 5.2.1 is applied to reject 2D/3D correspondences that do not provide consensus on the PnP pose solution. An example of 2D/3D correspondence generation between an image of the Flight Campaign 2 - Flight 5 sequence and the reference image used to initialize the landmark database is shown for the SIFT keypoint in Figure 25, and for the BRISK keypoint in Figure 26. The number of 2D/3D correspondences passing both the ratio test and the RANSAC outlier detection test are highlighted as lines between the flight image on the left of the figure, and the satellite reference image on the right. Keypoints that passed the ratio test, but failed the RANSAC test are represented as un-connected blue points.

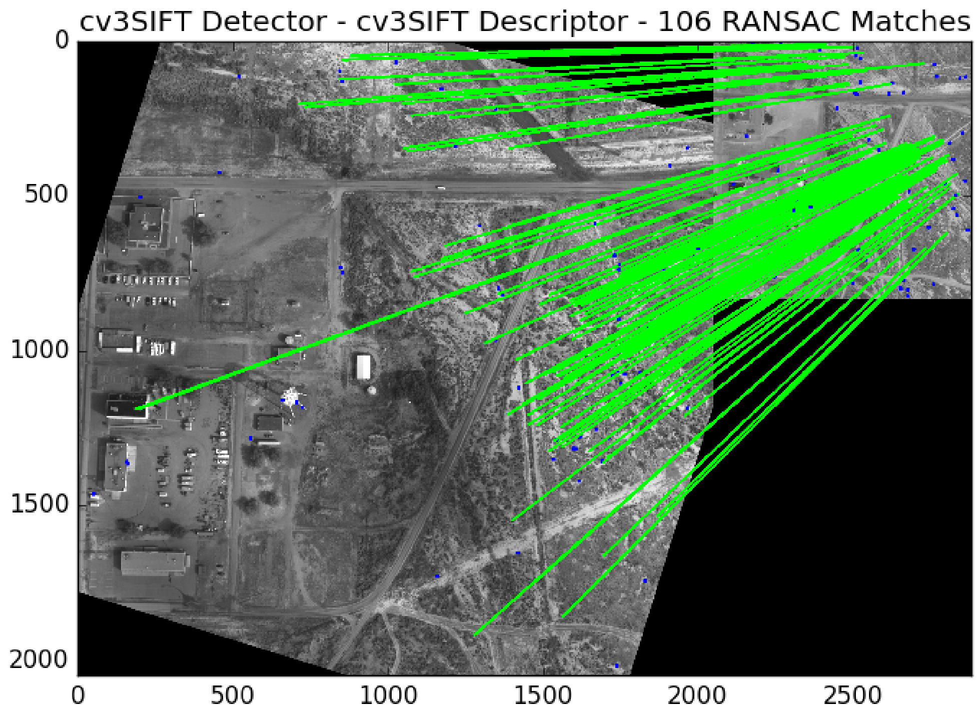


Figure 25: Example of 2D/3D correspondence generation using the SIFT keypoint and descriptor vector. Left image is a representative image from Flight Campaign 2 - Flight 5. Right image is a sub-image from the satellite reference image used to generate the landmark database. Reference satellite imagery used with permission from DigitalGlobe.

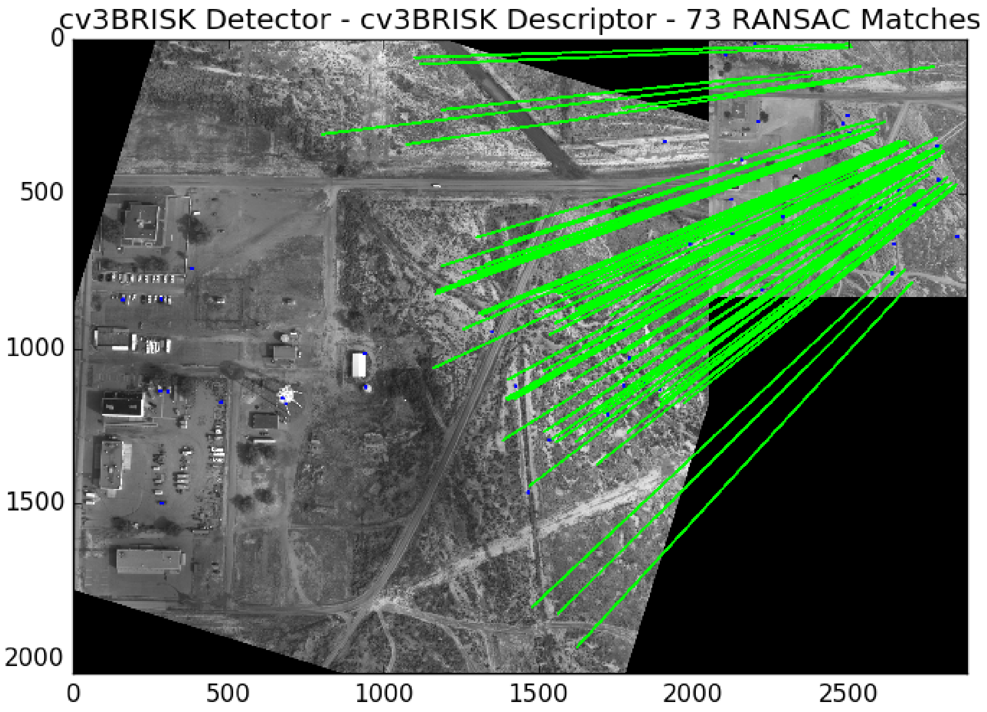


Figure 26: Example of 2D/3D correspondence generation using the SIFT keypoint and descriptor vector. Left image is a representative image from Flight Campaign 2 - Flight 5. Right image is a sub-image from the satellite reference image used to generate the landmark database. Reference satellite imagery used with permission from DigitalGlobe.

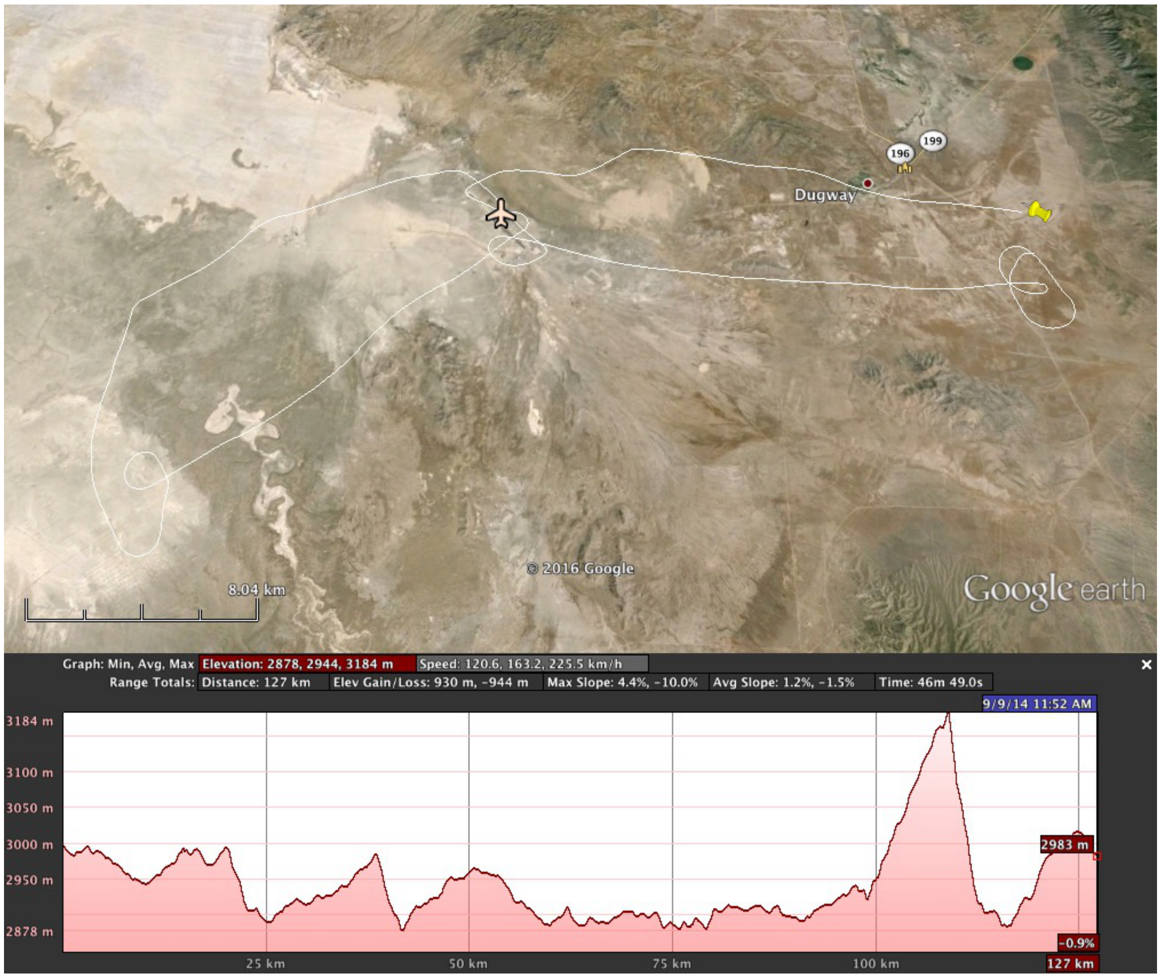
Table 5: Summary of Trajectories for Flight Campaign 2

Flight	Images Captured	Duration (min)	Min / Max WGS-84 Altitude (m)
1	1462	48.7	2849 / 2979
2	1074	35.8	2828 / 3054
3	1514	50.5	1932 / 2844
5	1349	45.0	2878 / 3185

5.5 *Experimental Results*

A series of flight test experiments were performed to validate the camera calibration and 3-DoF PnP algorithms described in this Chapter. For this experiment, 0.5m GSD reference satellite imagery provided by the National Geospatial-Intelligence Agency (NGA) to create 3 landmark databases for varying keypoint detector / descriptor types described in Chapter III: SIFT, BRISK, and SURF. The airborne camera used in this experiment was a Prosilica GE2040 machine vision camera, with a high-quality 50mm Canon EF-mount lens. The camera was rigidly mounted to a tactical-grade HG-1700 IMU. The experimental platform also utilized a Novatel OEM-V GPS receiver for initialization and truth reference. The GPS receiver hosted the Novatel SPAN firmware, which integrated the measurements from the GPS receiver and the HG-1700 to provide the truth-reference solution for the experiment. The equipment was integrated into a wing-mounted pod carried on a Cessna-172.

Three flight test campaigns were conducted for this research effort, the second of which provides the majority of the results for this research. During the second flight test campaign, 5 flights were conducted, 4 of which were used to generate the results shown in this Chapter. The flights followed roughly the same profile, and a representative trajectory (taken from Flight 5) is shown in Figure 27. Images were captured every other second (0.5Hz), with a nominal altitude of 2900m above the WGS-84 ellipsoid. The average height of the terrain in the area covered by the landmark database is 1363m above the WGS-84 ellipsoid, resulting in a nominal 1500m height above terrain. A summary of the 4 flights used to generate results is provided in Table 5.



Graph: Min, Avg, Max Elevation: 2878, 2944, 3184 m Speed: 120.6, 163.2, 225.5 km/h

Range Totals: Distance: 127 km Elev Gain/Loss: 930 m, -944 m Max Slope: 4.4%, -10.0% Avg Slope: 1.2%, -1.5% Time: 46m 49.0s

Figure 27: Top: Trajectory of Flight Campaign 2 - Flight 5. Bottom: Altitude profile and speed statistics.

The results of the auto-calibration process, and the subsequent PnP results are provided in the following sections.

5.5.1 Camera Calibration Results. This section provides the results of the automatic camera calibration routine performed using experimental data from Flight Campaign 2. Using the specifications of the Prosilica GE2040 camera, the intrinsic pinhole camera model was calculated as:

$$\mathbf{K} = \begin{bmatrix} 6756.75 & 0 & 1024.0 \\ 0 & 6756.75 & 1024.0 \\ 0 & 0 & 1 \end{bmatrix}$$

where the focal length was calculated by dividing the nominal focal length of the lens, 50mm by the pixel size of the Prosilica GE2040 camera, 7.4μm. The nominal principal point of the camera was calculated by assuming the z-axis of the camera frame intersected the image plane at the center of the image, in this case, $\mathbf{p} = [1024.0, 1024.0]^T$. From the mounting diagram, the camera frame was determined to be rotated 90° clockwise about the body frame, such that the x-axis of the camera points out the right wing of the aircraft, the y-axis points toward the rear, and the z-axis points down.

For each of the 1074 images in Flight 2, 5K of the best BRISK keypoints (sorted by response, and scale-filtered) were matched against 10K of the BRISK landmarks retrieved from the landmark database about the nominal projected center point of the image, as calculated using the truth-reference trajectory. For each set of initial matches, the 3-DoF PnP algorithm was used to identify inlier correspondences. Using the nominal camera calibration, 3-DoF PnP generated translation estimates with 3D-RSS position error roughly in the 50-150m range. From the 1074 images, a subset of 680 images were generated whose 3D-RSS error was less than 100m, and contained at least 20 inliers. From this subset, a further 250 were subsampled, in order to reduce the amount of computation needed to perform the camera calibration. This resulted in 27,191 2D/3D correspondences being used in the automatic camera calibration.

A plot of the residual projection error using the nominal camera calibration is shown in Figure 28. The statistics of the projection error conditioned on using the nominal calibration are given in the title of Figure 28, with the x-axis having a mean error of -22.91 pix, and a standard deviation of 39.43 pix. The y-axis mean is 95.66 pix, with a standard deviation of 36.85 pix.

The full-calibration state vector was then estimated using the iterative non-linear Least-Squares estimator. The resulting image-frame projection error after 29 iterations of the calibration process is shown in Figure 29. The residual vector from the calibrated camera parameters is zero mean, with 14.26 pix standard deviation in the camera x-direction, and 7.96 pix standard deviation in the camera y-direction. Within-image variance is much smaller, showing that there is still some image-dependent error in the system, potentially either local world point biases, or noise in the attitude estimate from the truth-reference system. The full calibration process also estimated a significant global bias for points in the world frame, $\mathbf{b}^w = [-1.259, 10.063, -4.692]^T$, where the world frame is the North-East-Down frame and units are in meters. This bias was then used to update the positions of the landmarks in the database. An illustration of this bias is shown in Figure 30. The push-pins in Figure 30 show the manually-surveyed camera calibration points. The red circles represent the projection of each calibration point onto the reference satellite image used to initialize the landmark database using the provided geographic metadata. The green circles represent the corrected locations after correcting for the bias vector, $\mathbf{b}^w$, which is represented by the blue line. The following sections show the results for the PnP algorithm for multiple flights, using multiple feature types, and shows the effect of using different camera calibrations.

5.5.2 Comparison of Camera Calibration Methods. This section overviews the results of using the data from Flight Campaign 2 - Flight 5 to compare the benefit

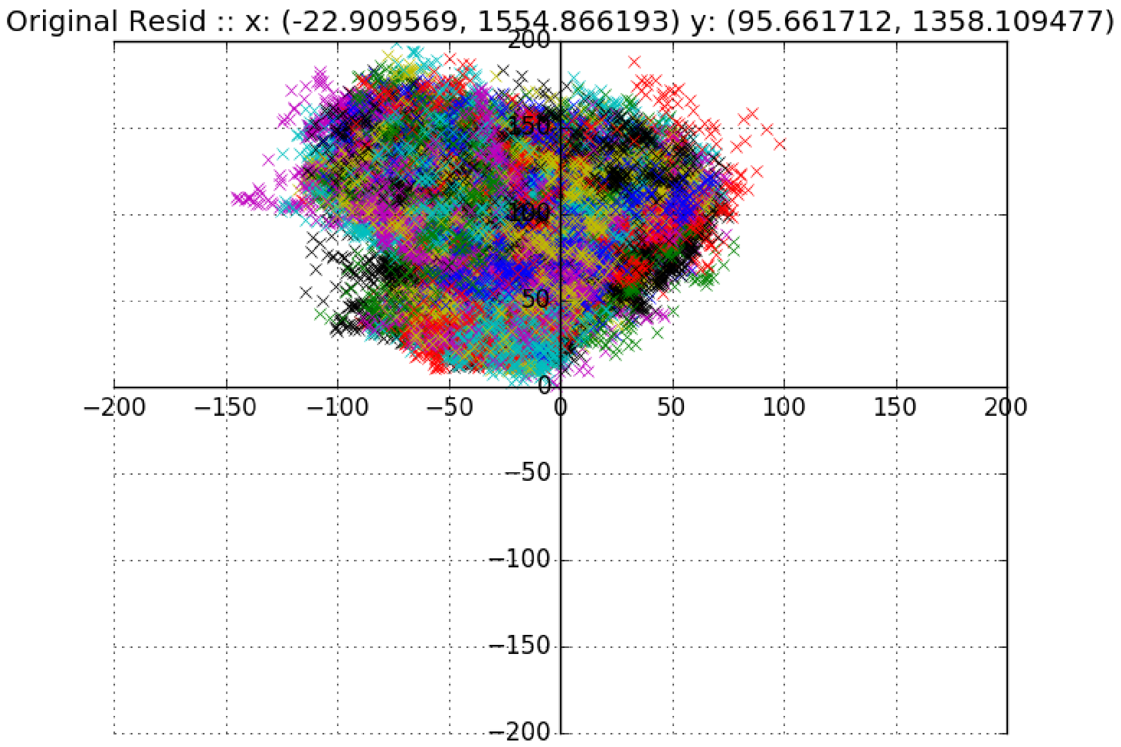


Figure 28: Initial image-frame projection residuals for 250 images sampled from Flight 2. The mean (pix) and variance (pix^2) of the image plane x and y residuals are provided in the title. Each color represents residuals from a specific image.

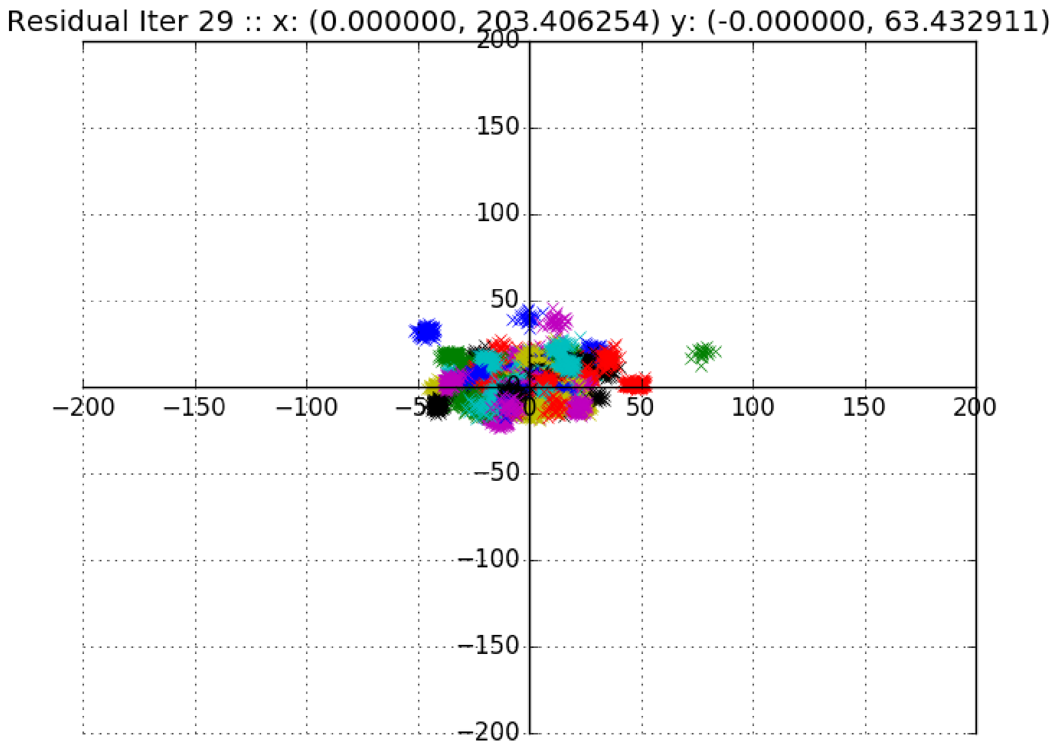


Figure 29: Final image-frame projection residuals for 250 images sampled from Flight 2. The mean (pix) and variance (pix^2) of the image plane x and y residuals are provided in the title. Each color represents residuals from a specific image.

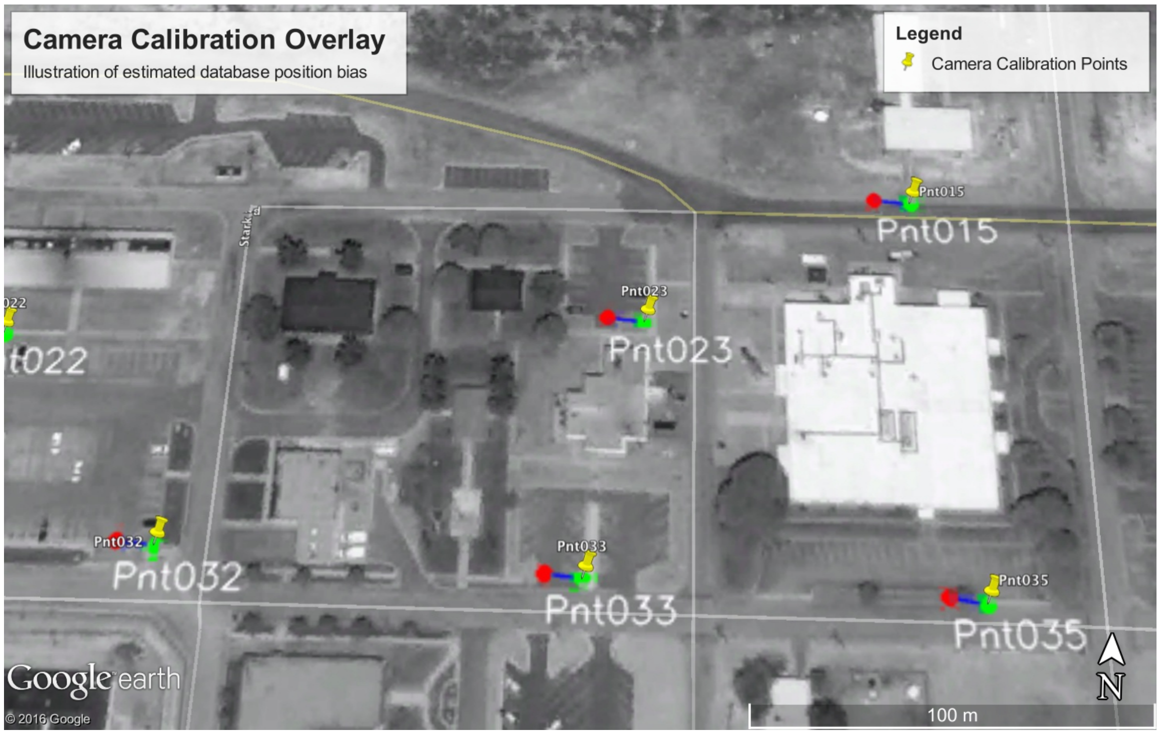

Camera Calibration Overlay
Illustration of estimated database position bias
Legend
Camera Calibration Points
Pnt015
Pnt023
Pnt022
Pnt032
Pnt033
Pnt035
Google earth
© 2016 Google
100 m
N

Figure 30: Illustration of landmark database position bias estimate. Push-pins represent the manually surveyed locations of calibration points. Red circles show the projection of these calibration points onto the reference satellite image used to initialize the landmark database, using the provided projection metadata. Green circles show the projection of the calibration points onto the reference image when corrected using the global bias estimate from the camera autocalibration algorithm.

of various camera calibration values. The 3DoF PnP algorithm was used to compute aircraft position for each image in Flight 5, using intrinsic and extrinsic camera calibration parameters computed by three different methods. The first camera calibration method, denoted as ‘Nominal Cal’, is a camera calibration computed using the nominal values of focal length, principal point, and rotation about the body frame from the camera and mounting specification, discussed in the previous section. The second camera calibration, denoted ‘Autocal - Camera Parameters Only’, was computed using the auto-calibration routine described in Section 5.3.2, without estimating the global bias of the landmark coordinates, using 2D/3D correspondences computed from Flight Campaign 2, Flight 2. The final calibration, ‘Autocal - Camera + Landmark Bias’, was computed using the same auto-calibration process as ‘Autocal’, this time estimating the global bias for the world coordinates in the landmark database.

The 3D RSS error for the 3DoF PnP pose estimate using the BRISK keypoint detector and descriptor vector is shown in Figure 31. The error is decomposed into north, east, and down components, and shown in Figure 32. The nominal calibration provides consistently biased results across the entire flight trajectory, with a final mean value of 64.45 m. The ‘Autocal’ calibration provides significantly better results, however still is biased, with 3D RSS mean value for the flight of 14.15 m. When the global offset for all the landmark coordinate is applied to the landmark database using the estimate from the ‘Autocal - Camera + Landmark Bias’ calibration, the RSS error is reduced to 5.6 m, showing most of the error to be contained in the down direction. This indicates that there may be some error either due to some non-uniform height bias in the landmark database, or the focal length estimate of the camera.

These results demonstrate the ability to use the landmark database, along with a nominal camera calibration to produce relatively accurate estimates of aircraft position in subsequent flights, therefore removing the need for analyst/survey intervention when implementing vision-aided navigation in a flight environment. The subsequent results in this Chapter use the parameters from the ‘Autocal - Bias’ camera calibration, with the estimate of the global landmark position bias having been accounted for in the landmark database.

5.5.3 PnP Results Using Different Keypoints. This section examines the performance of the PnP pose estimation algorithm across multiple keypoint detectors and descriptor vectors. Again, Flight 5 from Flight Campaign 2 was used as the evaluation trajectory. Three different keypoint detector and descriptor vectors were evaluated: BRISK, SIFT, and SURF. The OpenCV 3.1.0 implementation of each of these keypoint detectors and descriptor extractors were used for this evaluation.

The first metric evaluated was the total number of images that generated a PnP solution, conditioned on the keypoint detector used. To place this metric in context, two additional plots related to the number of keypoints generated by each detector are provided. First, Figure 33 illustrates the number of images that generate at least the $j = 5\text{K}$ number of keypoints. Of the 1349 images captured during Flight 5, Figure 33 shows that 950 of the images generated at least 5K BRISK keypoints, with 1052 images for SIFT, and 1158 for SURF.

The distribution of the number of keypoints generated per image for all three detectors is shown in Figure 34. For Flight 5, the SURF detector generates more keypoints than either BRISK or SURF. This makes the result shown in Figure 35, the number of images in Flight 5 that generate a pose estimate from PnP, somewhat counter-intuitive. While SURF generates, on average, more keypoints per image, the PnP pose estimation using SURF results in far fewer resulting pose estimates than BRISK or SIFT. Figure 36 shows the number of keypoints per image that passed the RANSAC outlier detection test, illustrating that SURF produces fewer inliers per image. While SURF produces more keypoints on average than BRISK or SIFT, the use of SURF keypoints in pose estimation results in fewer position solutions.

The next results examine the accuracy of the position solution estimated by the 3DoF PnP algorithm during Flight 5, conditioned on the 3 keypoint detectors. Figure 37 provides a box plot of the 3D RSS error for the 3DoF PnP pose estimate

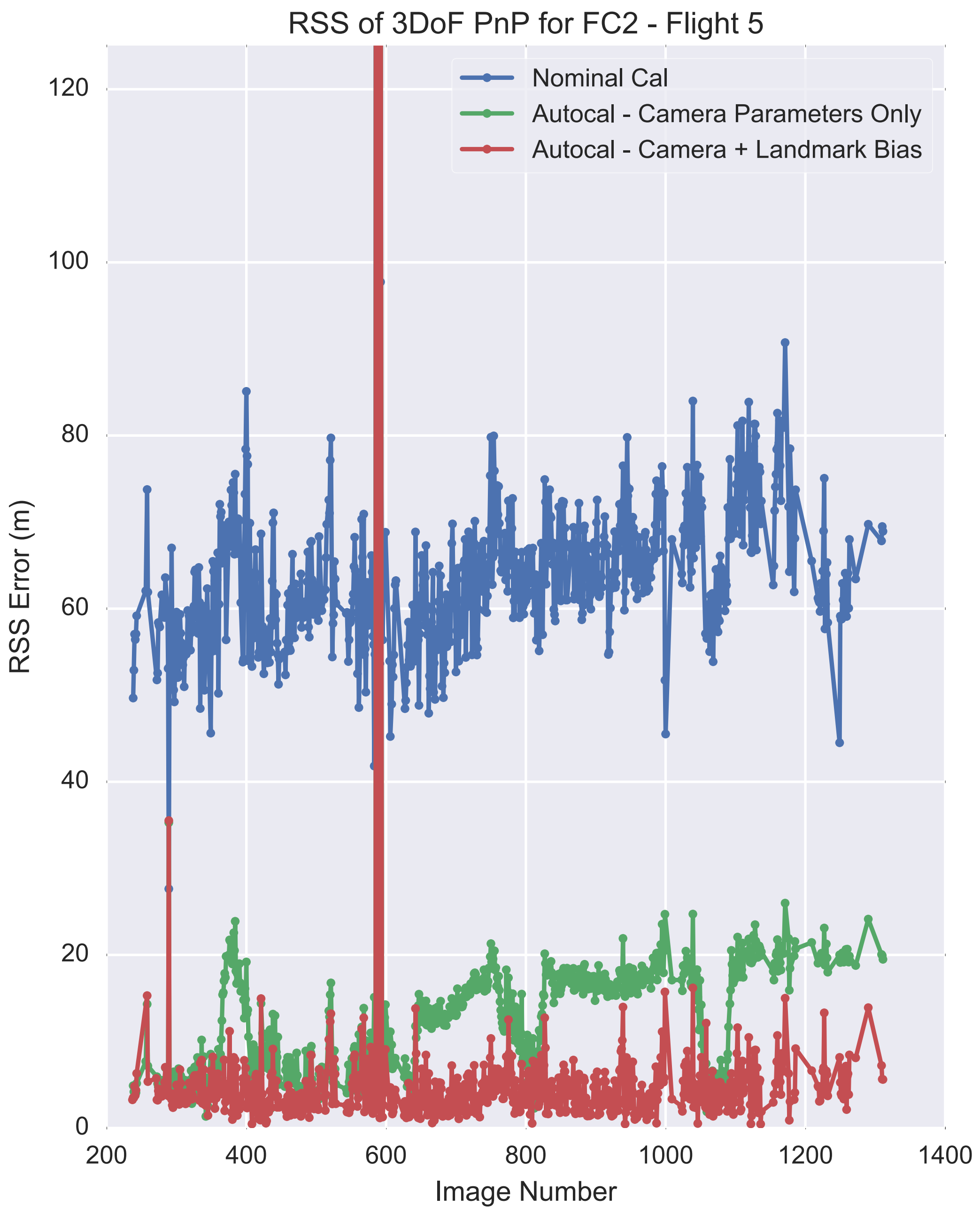


Figure 31: 3D Root Sum of Squares error for 3 degree-of-freedom PnP solution for Flight Campaign 2, Flight 5 for multiple camera calibration values.

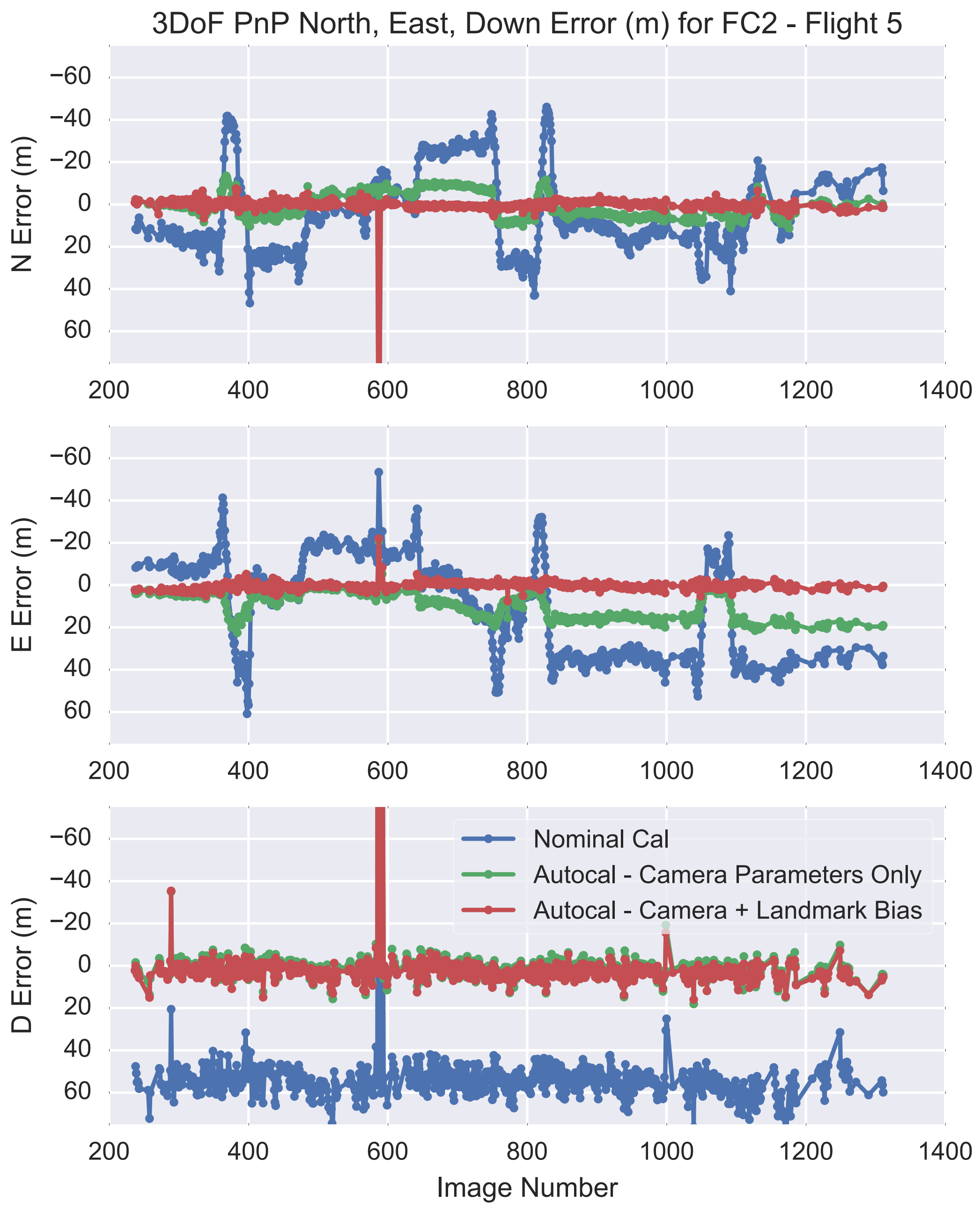


Figure 32: North, east, and down error for 3 degree-of-freedom PnP solution for Flight Campaign 2, Flight 5 for multiple camera calibration values.

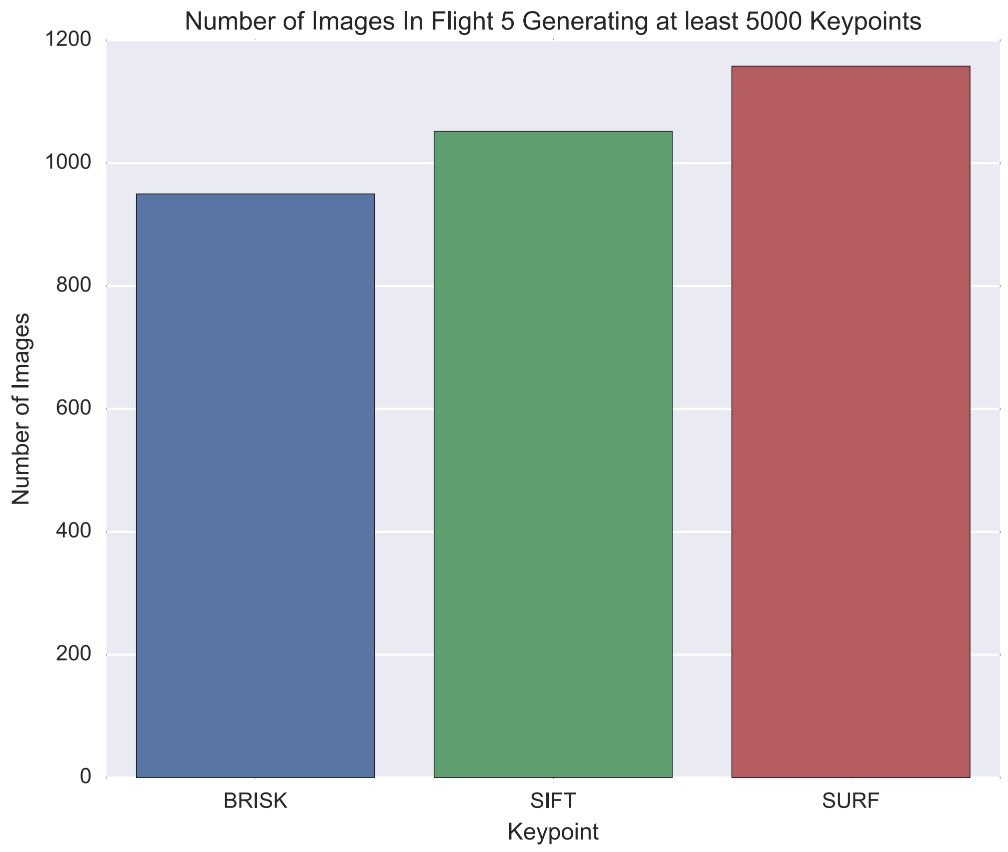


Figure 33: Number of images in Flight Campaign 2, Flight 5 which generated over 5000 keypoints using multiple keypoint detectors.

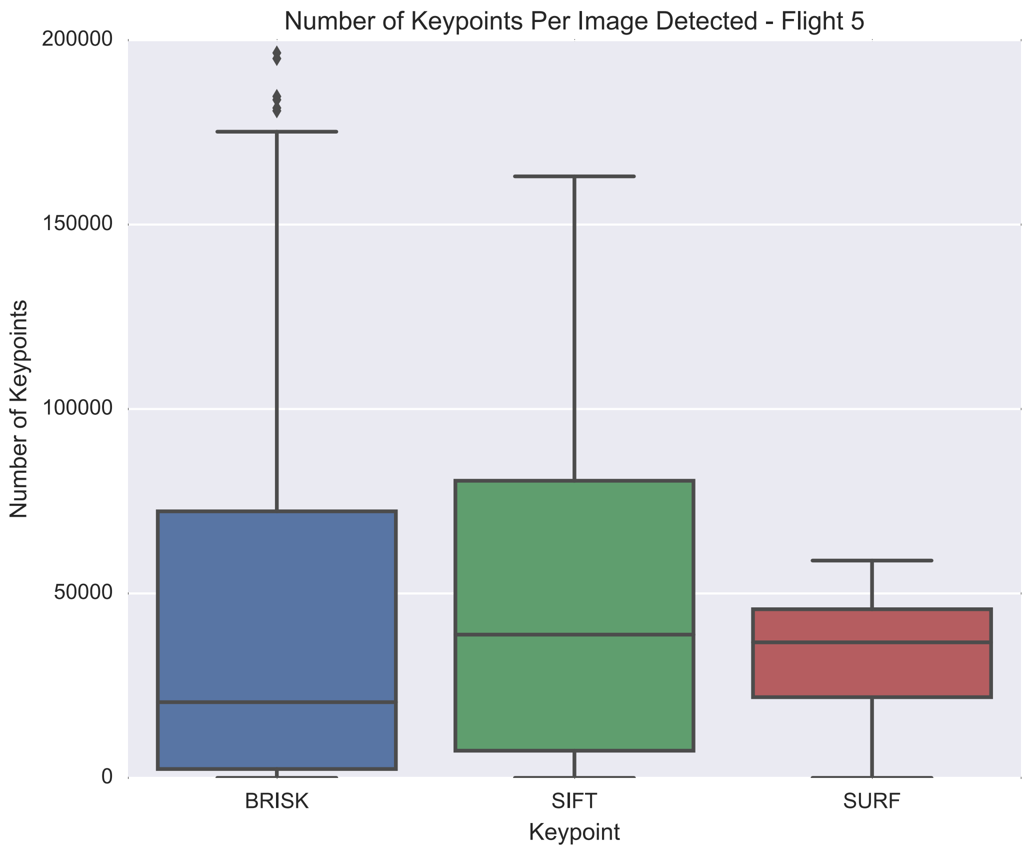


Figure 34: Box plot illustrating the distribution of the number of keypoints generated per image during Flight Campaign 2, Flight 5. Box plots illustrate a distribution by segmenting quartiles of the data, where each component of the colored box representing 25% of the data, and the area between the edge of the colored box and the whisker representing another 25%. The solid line within the colored box shows the median value of the data. Data determined to be outliers (1.5 times the interquartile range) are represented by diamonds.

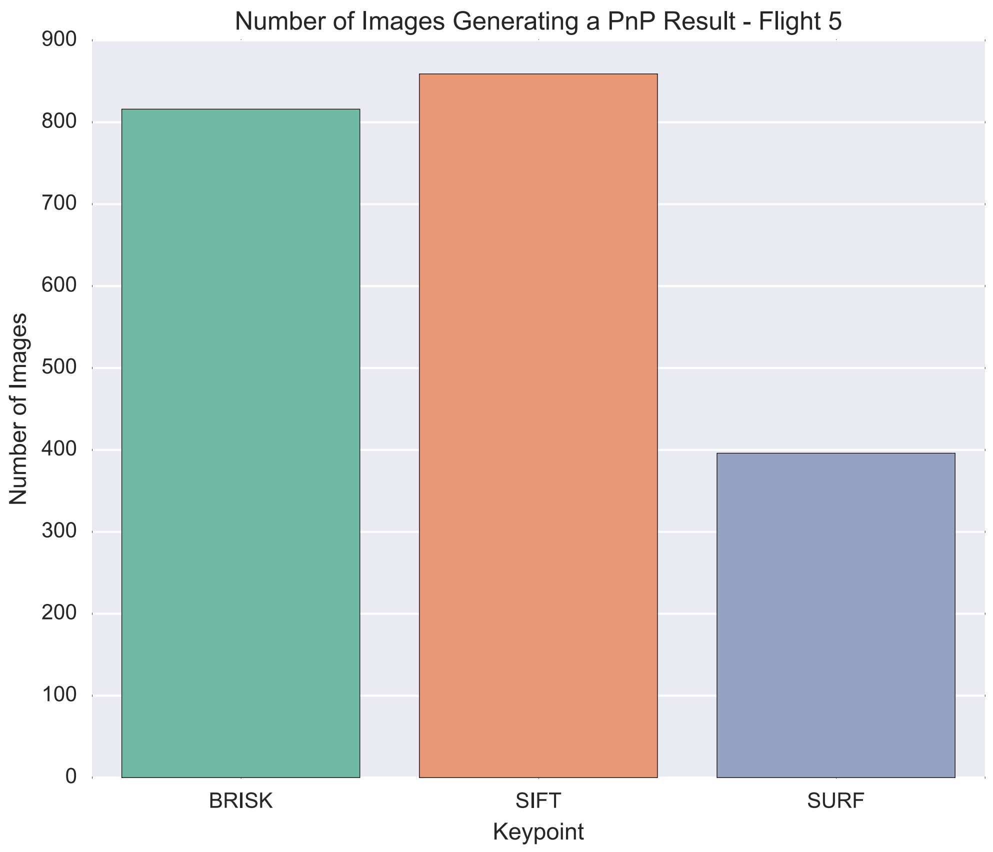


Figure 35: Number of images in Flight Campaign 2, Flight 5 which generated a pose estimate from PnP.

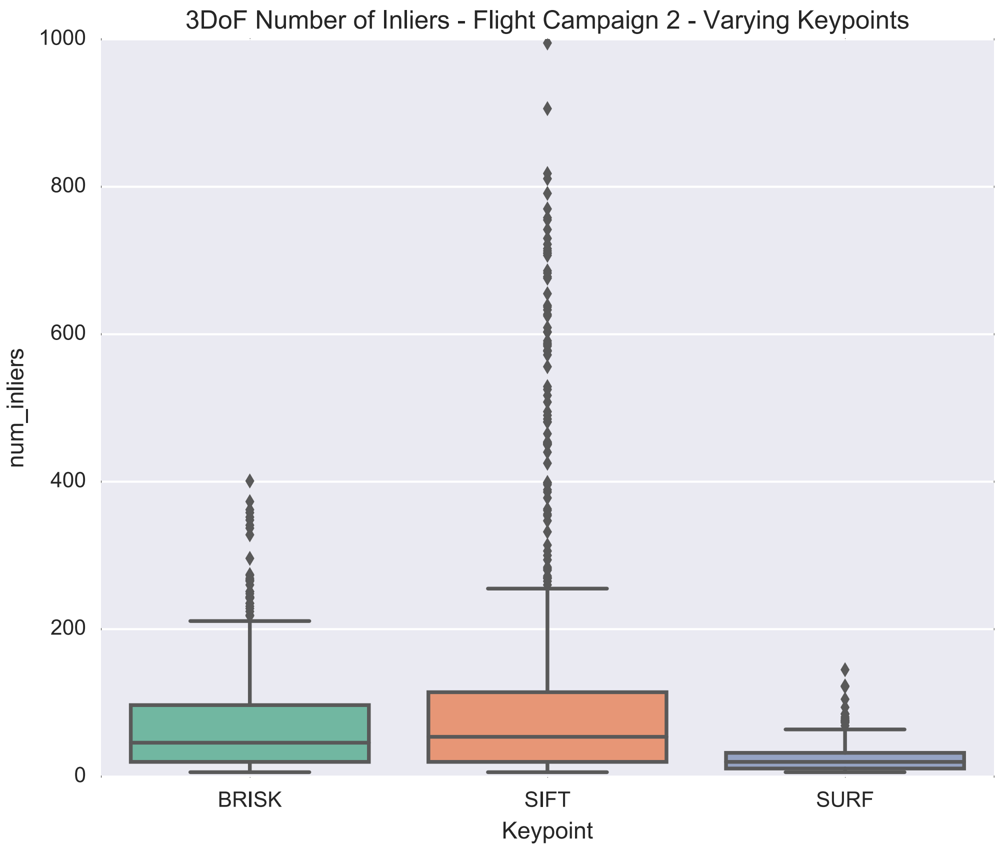


Figure 36: Distribution of the number of inliers per image that result in a the generation of a position estimate from PnP, during Flight 5.

over the Flight 5 trajectory, using BRISK, SIFT, and SURF keypoints and descriptors. Figure 38 shows the same information, but y-axis limited to show the detail of the quartiles of 3D RSS error.

A few images using the BRISK and SURF keypoint detectors and descriptor vectors generated pose estimates with extreme 3D RSS error, denoted as diamonds in Figure 37. Ignoring the few outliers, the distribution of 3D RSS error was relatively constant across the different keypoints, with 3D RSS error median values monotonically increasing from BRISK, to SIFT, and SURF estimates of aircraft position being slightly worse.

Therefore, with SURF providing significantly fewer estimates and slightly higher 3D RSS error, the following results focus on BRISK and SIFT. The majority of the results focus on the BRISK keypoint detector and descriptor, as each of the detection, description, and matching steps are significantly faster when using BRISK. In addition, the BRISK descriptor uses 16 times less storage, making it attractive for use in an ensemble analysis.

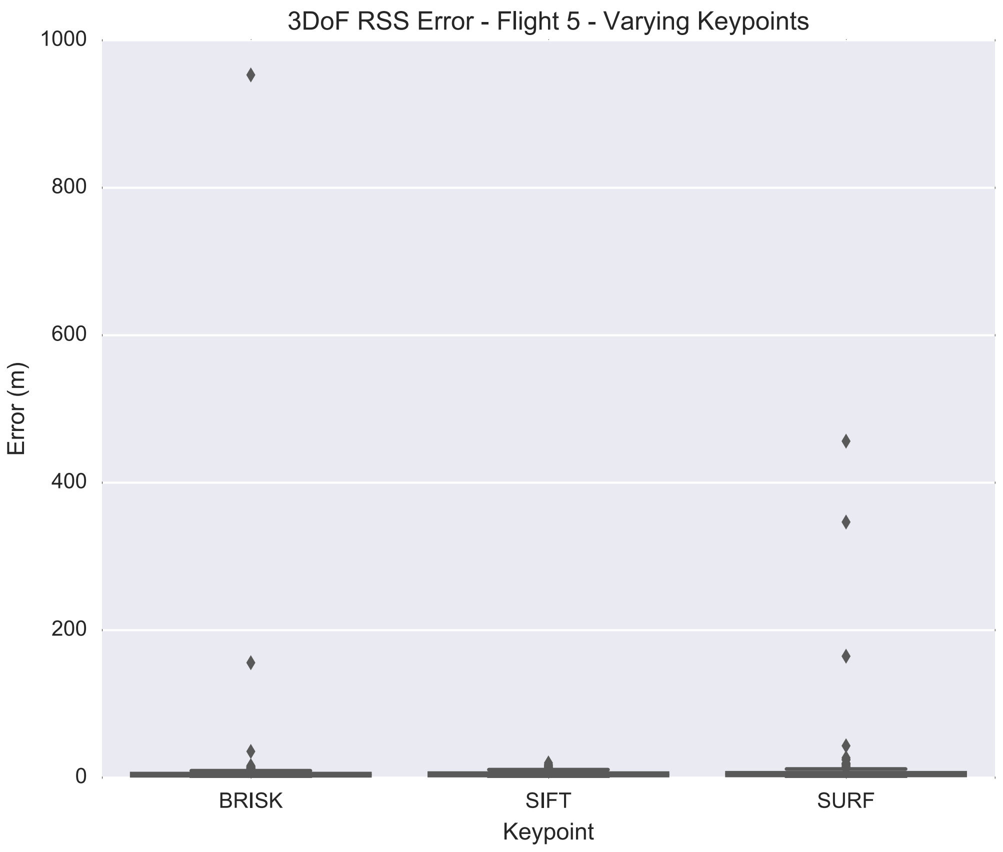


Figure 37: Distribution of the 3D RSS error for 3DoF PnP for Flight 5, compared against BRISK, SIFT, and SURF keypoint detectors and descriptor vectors.

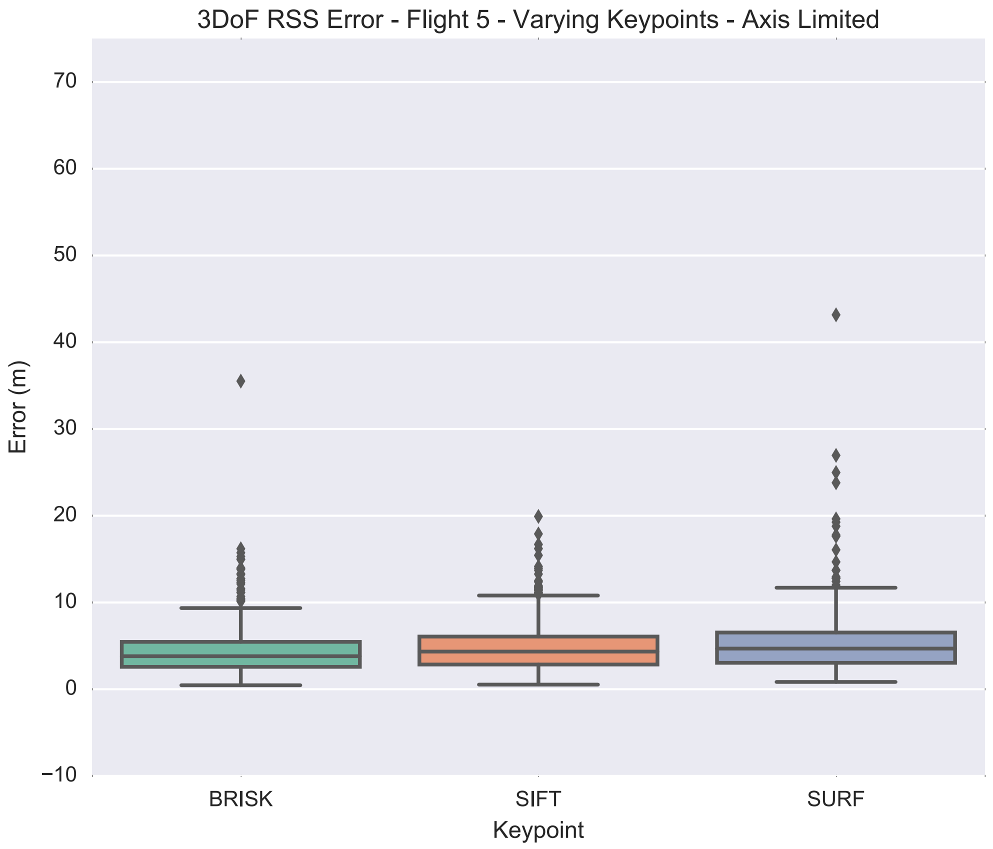


Figure 38: Y-Axis limited distribution of the 3D RSS error for 3DoF PnP for Flight 5, compared against BRISK, SIFT, and SURF keypoint detectors and descriptor vectors.

5.5.4 PnP Results Using BRISK. The following section details the results of the PnP position estimation algorithm when using the BRISK keypoint detector and descriptor extractor. As shown in the last section, PnP estimates using BRISK were shown to provide accurate estimates when compared to SIFT or SURF, and generate a similar number of results. The reduced computation and storage requirements for using BRISK were attractive for this analysis, which includes results from Flights 1, 3, and 5 of Flight Campaign 2. As Flight 2 was used in the camera calibration process, results using Flight 2 data were excluded from this analysis.

The first metric being presented is the reduction in error when using the 3DoF + INS attitude formulation of PnP as compared to the the full 6DoF PnP position estimates. Figure 39 shows a violin plot of the north, east, and down error for both the 6DoF (green) and 3DoF + INS (orange) formulations of PnP for 2237 images over Flights 1, 3, and 5. An axis limited version of the same result is shown in Figure 40.

The result demonstrates that both 6DoF and 3DoF PnP generate roughly zero-mean estimates of position in the horizontal directions, and slightly biased in the vertical direction (14.6m mean error for 6oF, 3.09m mean error for 3DoF). However, the distribution of error for each direction in the 6DoF case is significantly larger than the 3DoF error. For the horizontal cases, the entire distribution of 3DoF error (excluding outliers) falls within the first two quartiles of the 6DoF horizontal error. On average, the 3DoF PnP is nearly 15 times more accurate than 6DoF in a 3D RSS sense.

Further analysis of the 3DoF error shows that PnP is able to consistently generate accurate results of position. Figure 41 shows a box plot of the north, east, down error for Flights 1, 3, and 5 when using the 3DoF PnP estimator with BRISK keypoints. With the exclusion of one outlier in the down direction, most of the error is completely bound within 200m. Limiting the axis to $(-20, 20)$ in Figure 42, shows that the majority of the error for the horizontal directions is bound within 5m, and the vertical within 10. The north and east directions are nearly zero mean (-0.30m and 0.47m, respectively), with the down error being slightly biased with 3.09m mean error.

A geohistogram of the average 3D RSS error is provided in Figure 43. The results are binned into Spherical Mercator tiles with $z = 15$, based on the location of the center point of the image being used to generate the result. Transparent tiles represent areas of the landmark database that were observed, but no PnP estimate was generated. Notice that the majority of these tiles were generated near either salt-flat in the Northwest quadrant of the database, or riverbed in the Southeast.

Additionally, inspection of the approximate Hessian after the PnP problem has finished iterating provides an estimate of the covariance of the position solution, resulting from the geometry of the 2D/3D correspondences. Figure 44 provides an illustration of the north, east, and down error from the 3DoF BRISK PnP algorithm evaluated for Flight 5, along with the estimated 1σ bounds. Overall, 33% of the horizontal error fell within the 1σ, with 63% of the vertical error being contained. It appears that there is not a clear correlation between the estimated covariance estimates and the actual error, indicating that there are other more dominant error sources remaining to be resolved within the PnP pose estimation algorithm.

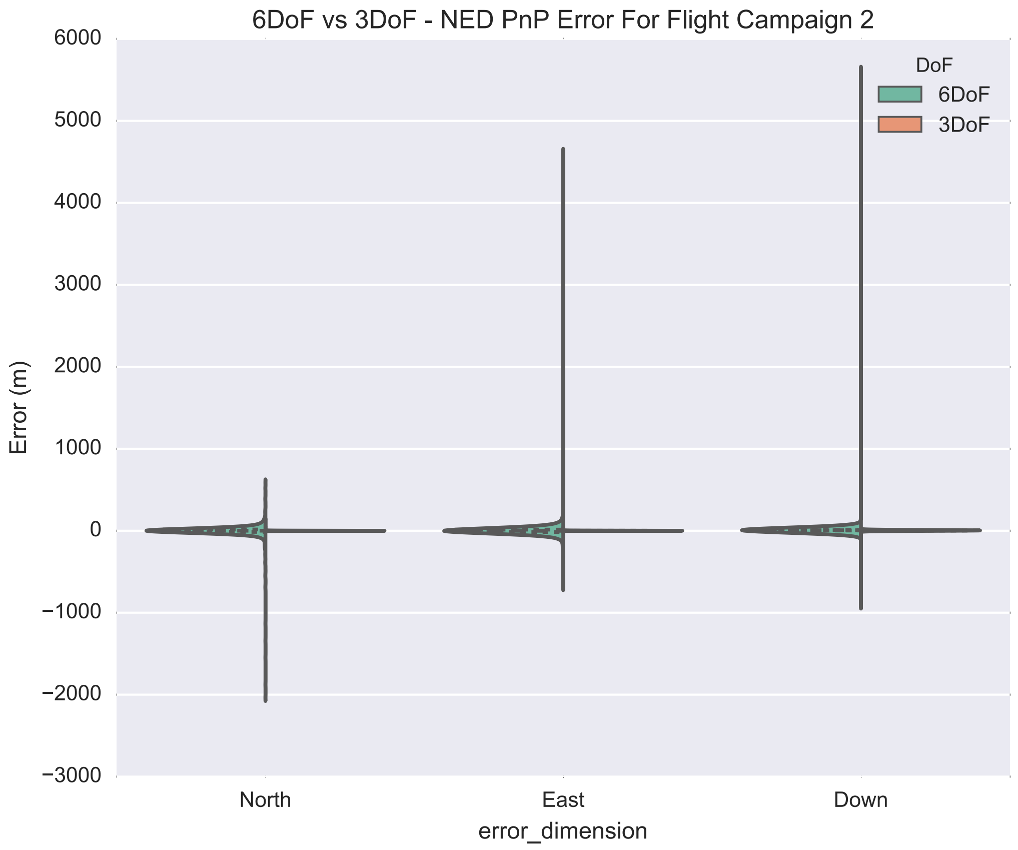


Figure 39: Violin plot comparing the north, east, down error between 6DoF and 3DoF PnP. Distribution uses 2237 PnP position estimates from Flights 1, 3, and 5. Inner dotted-lines on each half of the violin denote quartiles. The outer shape of the violin is an estimate of the distribution of the error in each direction.

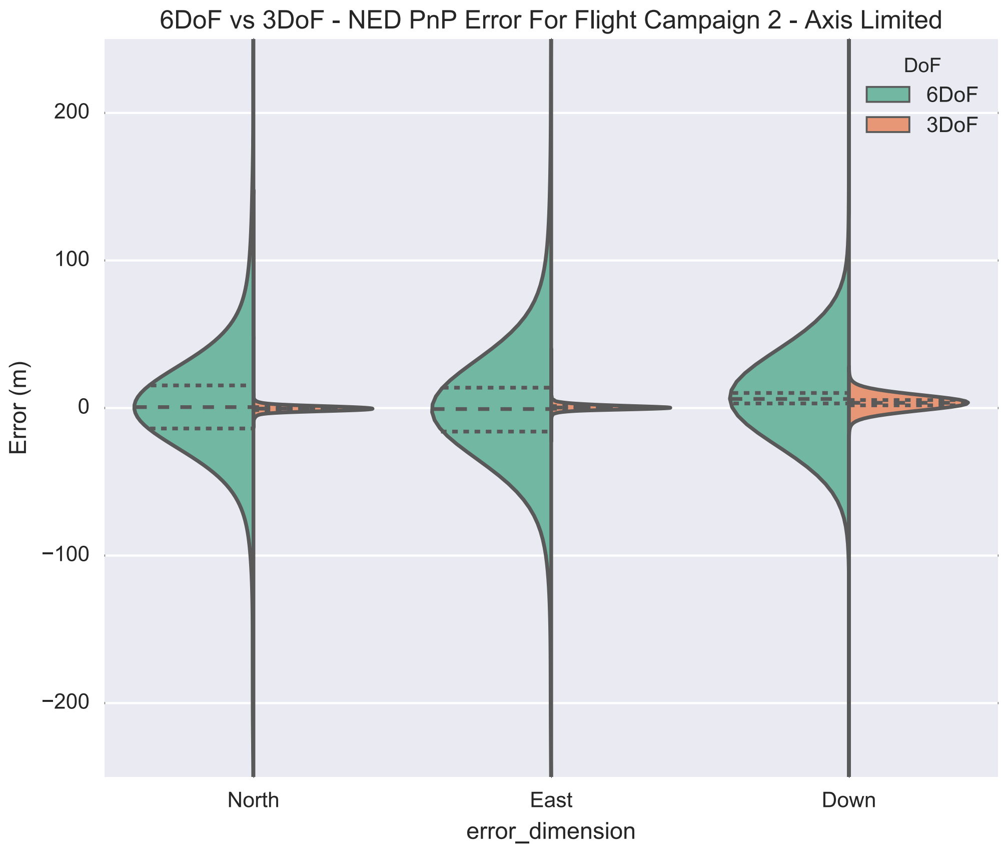


Figure 40: Axes limited violin plot comparing the north, east, down error between 6DoF and 3DoF PnP. Distribution uses 2237 PnP position estimates using BRISK keypoints from Flights 1, 3, and 5. Inner dotted-lines on each half of the violin denote quartiles. The outer shape of the violin is an estimate of the distribution of the error in each direction.

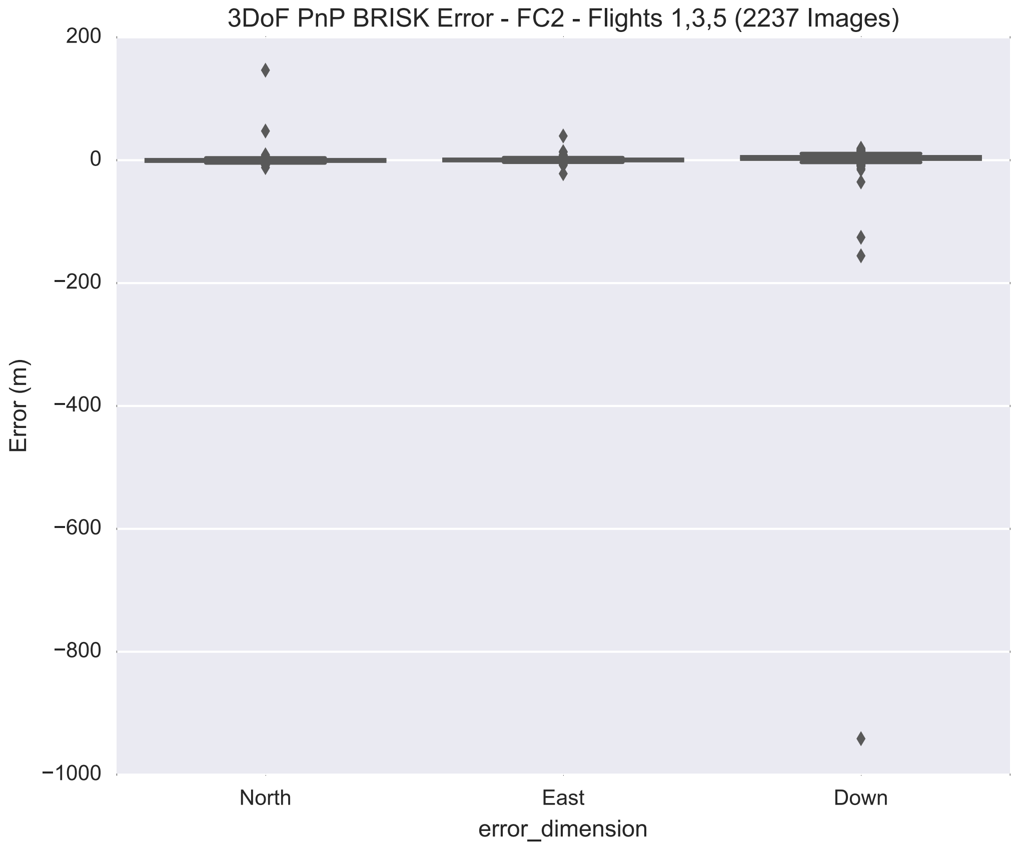


Figure 41: Box plot of north, east, and down error resulting from 2237 estimates of position from 3DoF PnP from Flights 1, 3, and 5.

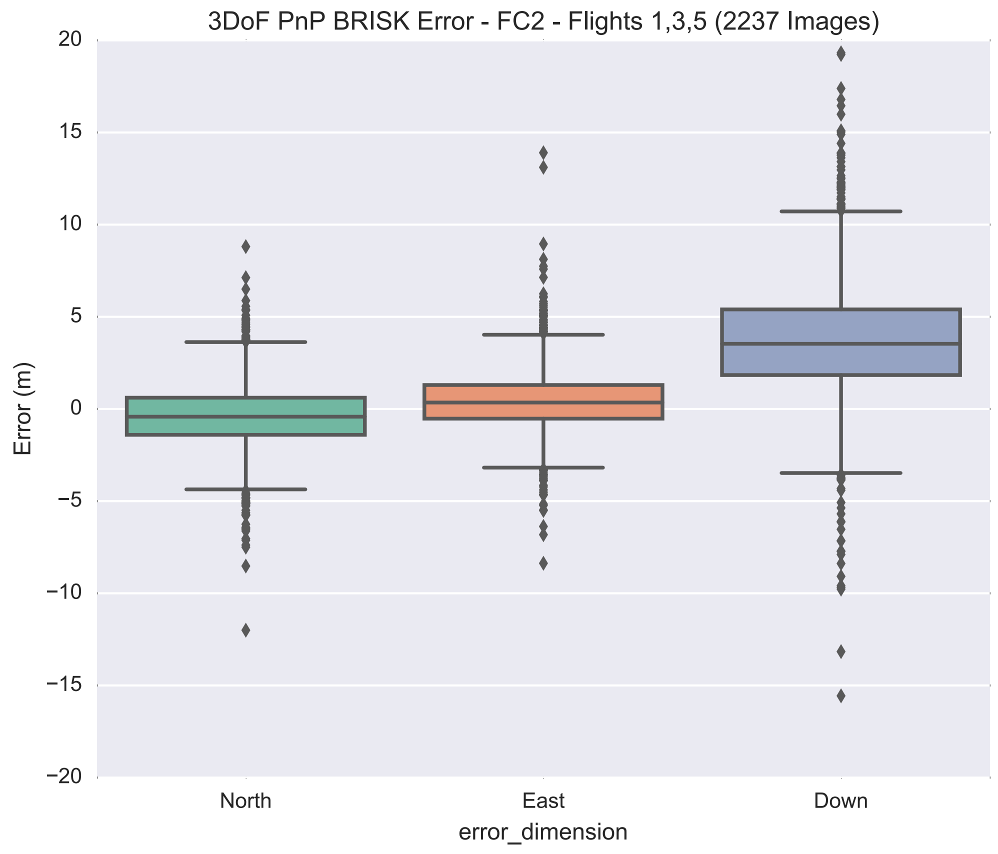


Figure 42: Axis limited box plot of north, east, and down error resulting from 2237 estimates of position from 3DoF PnP from Flights 1, 3, and 5.

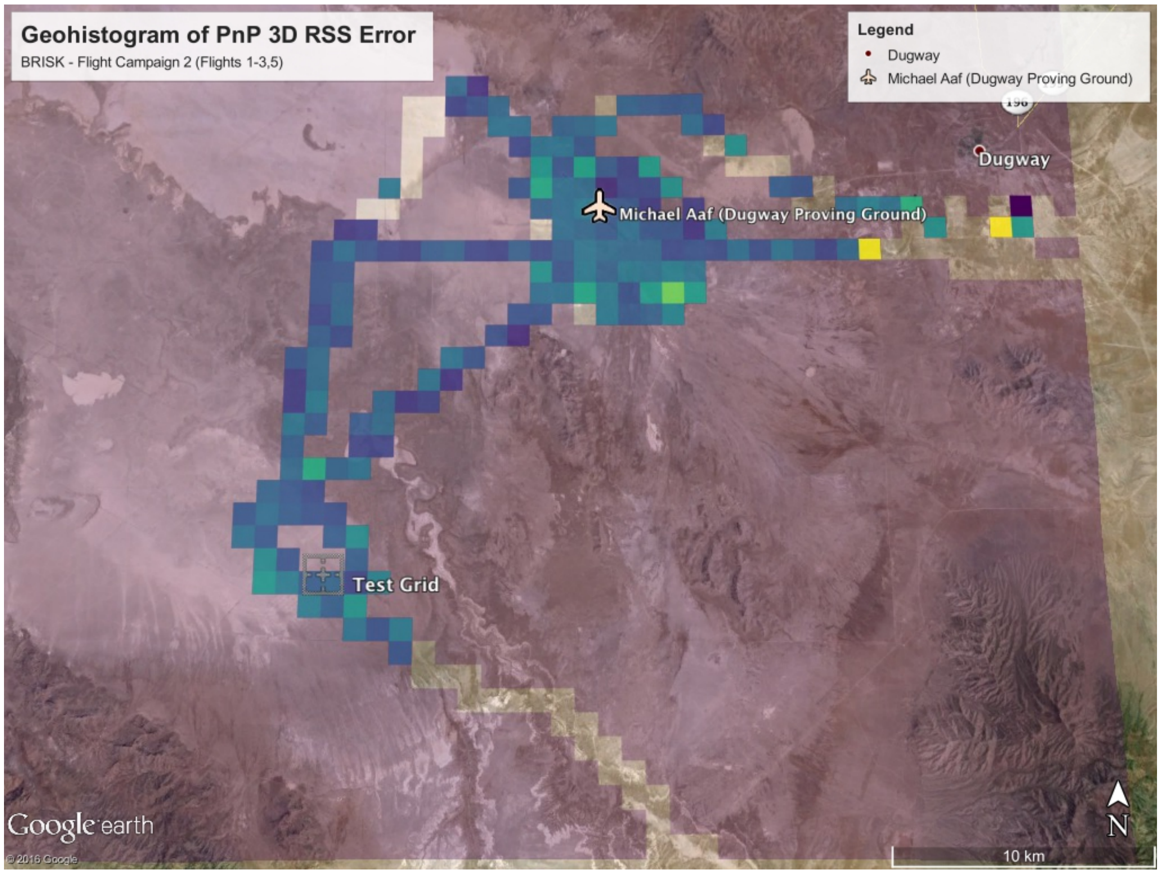

Geohistogram of PnP 3D RSS Error
BRISK - Flight Campaign 2 (Flights 1-3,5)
Legend
Dugway
Michael Aaf (Dugway Proving Ground)
Dugway
Michael Aaf (Dugway Proving Ground)
Test Grid
Google earth
© 2016 Google
10 km
N

Figure 43: Geohistogam of average 3D RSS error from 3DoF PnP position estimates over all flights in Flight Campaign 2. Transparent tiles represent areas of the landmark database that were observed, but no PnP estimate was generated. Light blue represents 3D RSS error of 0.0, with the maximum being shown in yellow at 35.5m

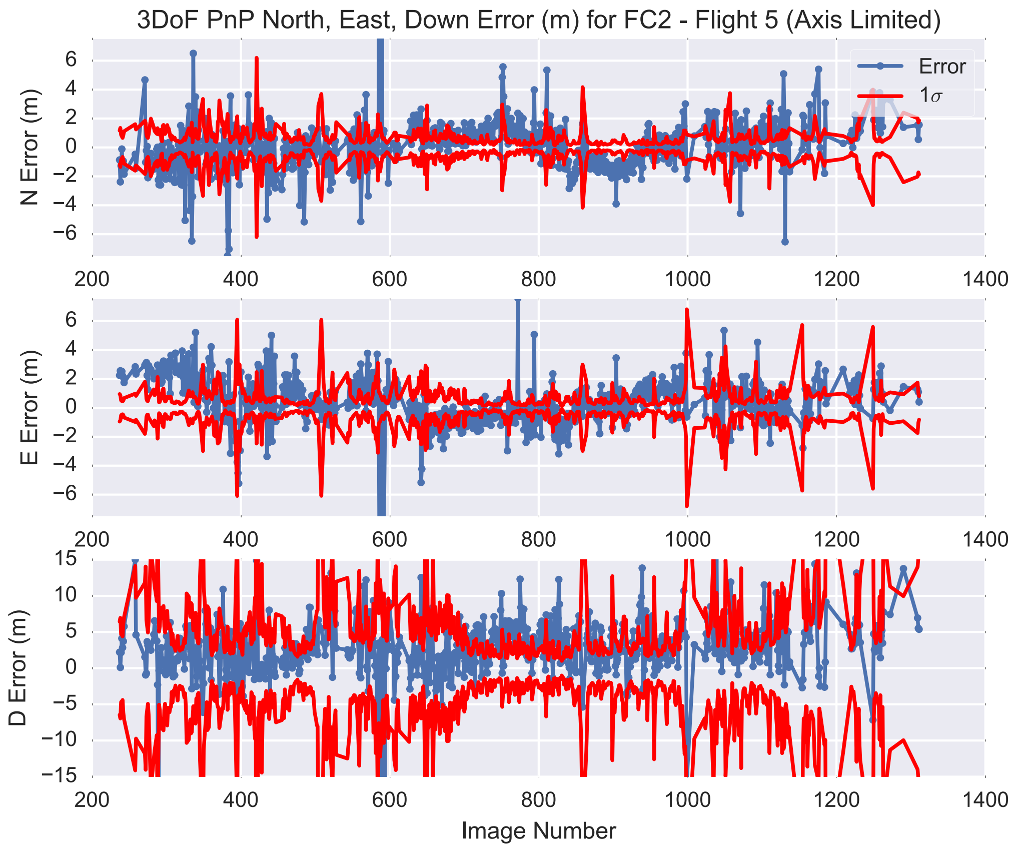


Figure 44: North, east, and down Error of 3DoF PnP (BRISK) with estimated 1σ bounds.

5.5.5 PnP Results Using Scale Filtering. The final section of results examines the use of the scale decomposition of both the observed image, along with the landmark database in order to improve 2D/3D correspondence generation. These results again examine Flight 5, comparing both the SIFT and BRISK keypoints. For the following results, two classes are identified. 'Scale Filtered' denotes that keypoints from the observed image whose absolute scale were smaller than the smallest absolute scale provided by the landmark database were discarded. In the Flight Campaign 2 case, the observed images when projected onto a terrain model generated an average Ground Sample Distance of 0.23m. The base layer of the landmark database was generated from reference imagery whose GSD was 0.5m. Therefore, all keypoints from the observed images from the first octave in the scale decomposition were discarded in the 'Scale Filtered' case. The 'All Scales' case represents results that were generated by matching keypoints from the observed image without filtering out keypoints from the first octave.

The first metric analyzed is the number of images that successfully generated an estimate of aircraft position using PnP. Figure 45 shows the number of images from Flight 5 that generated PnP position estimates using both BRISK and SIFT, conditioned on the use of scale filtering. Scale filtering was successful in generating slightly more PnP position estimates, with a slightly greater benefit for SIFT keypoints.

To help analyze these results, the following metrics are composed of statistics of the number of inlier 2D/3D correspondences. Figure 46 illustrates the total number of inlier 2D/3D correspondences for Flight 5, for both keypoints, conditioned on the use of scale filtering. The distribution of inliers for both keypoints shows that more inliers were generated for the scale filtering case. The next metric, shown in Figure 47, is the percentage of inliers as compared to the total number of 2D/3D correspondences returned by the matching ratio test. Again, scale filtering significantly increases the percentage of inliers to outliers for both BRISK and SIFT. The percentage increase is much greater for SIFT.

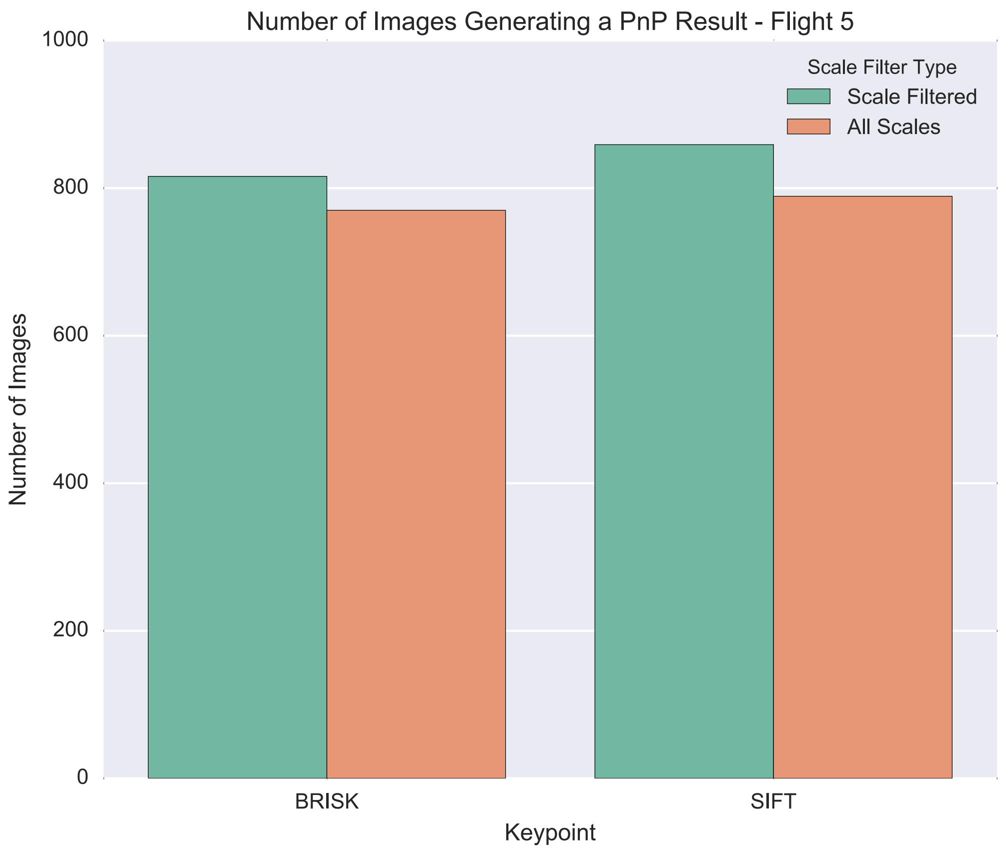


Figure 45: Number of images from Flight 5 returning a position estimate from PnP conditioned on keypoint and the use of scale filtering.

These results demonstrate that the use of scale filtering increases the total number of position estimates returned by PnP by increasing the percentage of inlier 2D/3D correspondences presented to the PnP solution. While scale filtering increases the availability of the PnP solution, Figure 48 shows that the overall accuracy of the solution is similar between the two cases.

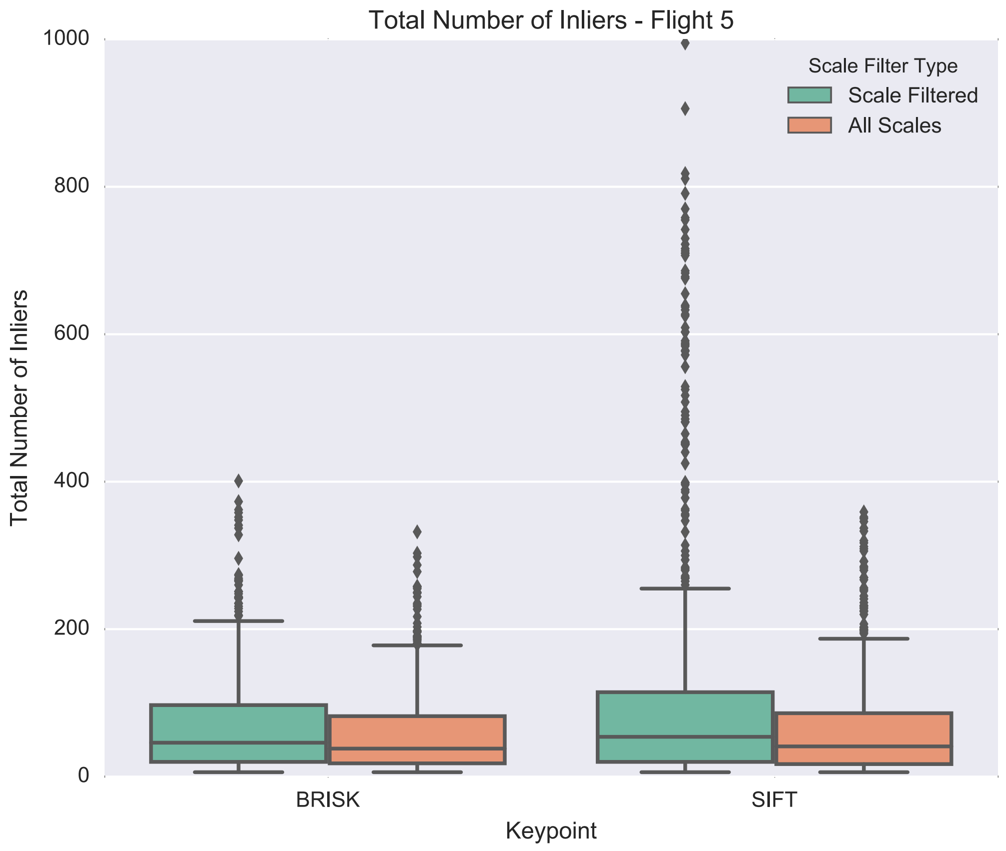


Figure 46: Box plot illustrating the distribution of the number of inlier 2D/3D correspondences from Flight 5 conditioned on keypoint and the use of scale filtering.

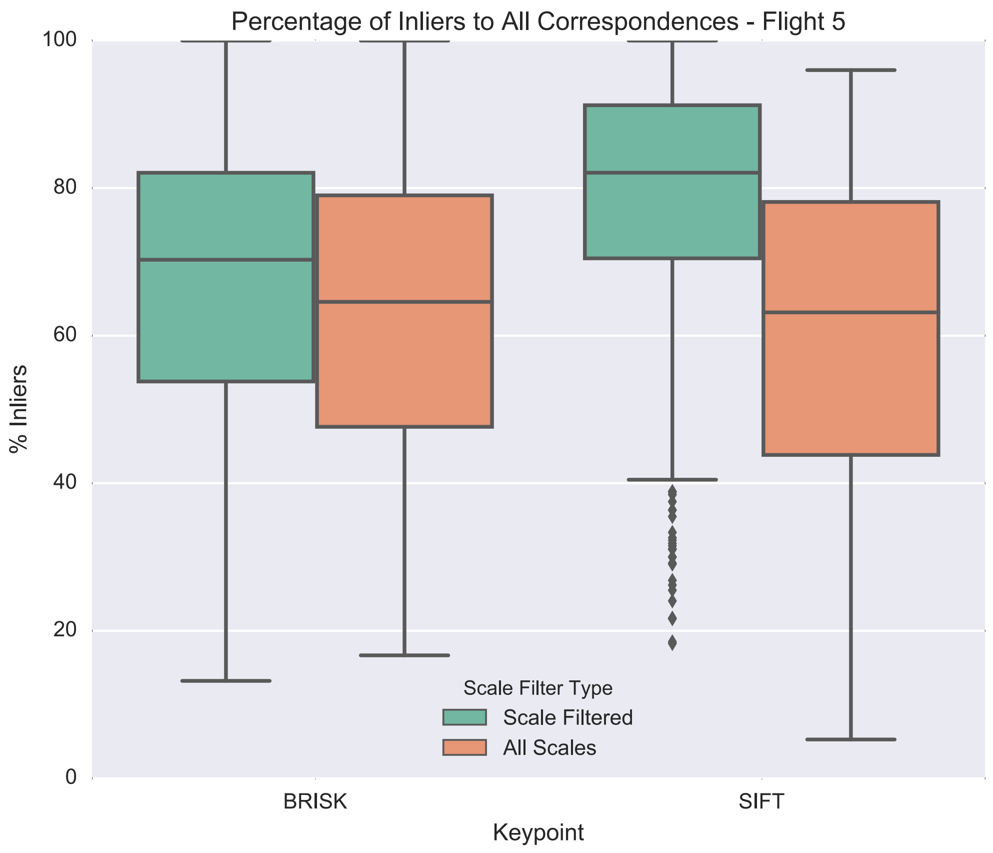


Figure 47: Box plot illustrating the distribution of the percentage of inlier correspondences as compared to the total number of 2D/3D correspondences from Flight 5 conditioned on keypoint and the use of scale filtering.

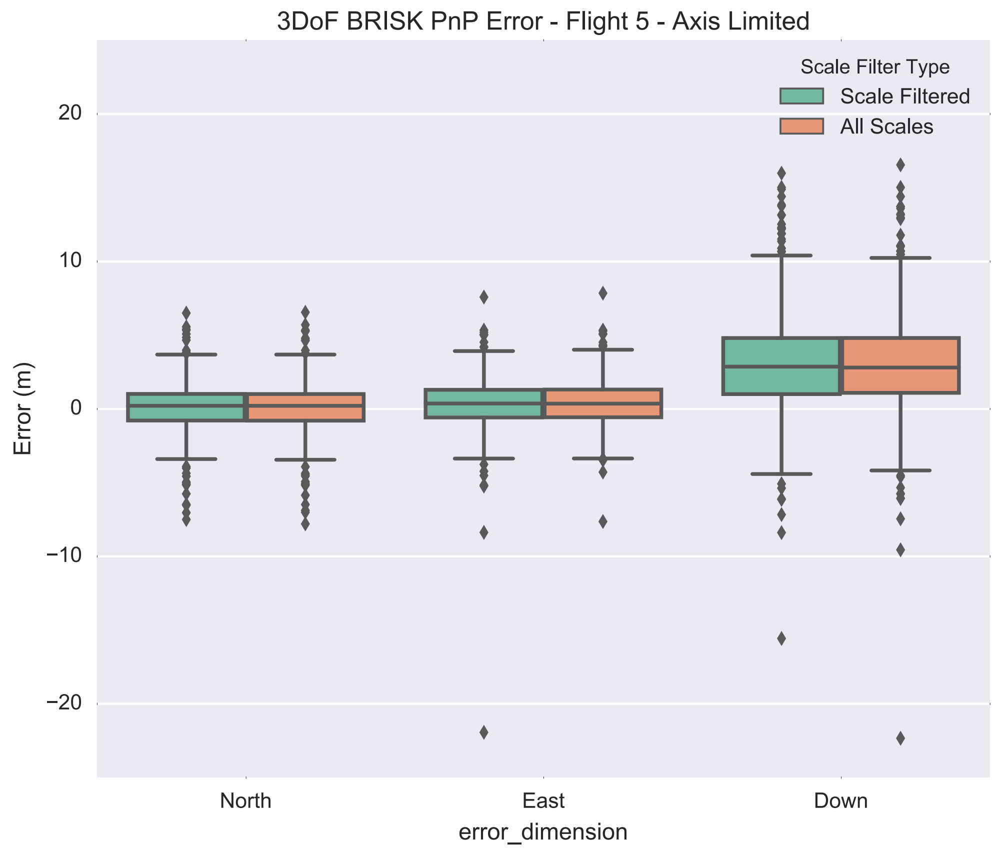


Figure 48: Box plot of north, east, down error for Flight 5 conditioned on scale filtering, for both SIFT and BRISK keypoints.

5.6 Chapter Summary

In this chapter, a method for estimating position using correspondences between keypoints detected in an image and a landmark database was derived. This research optimized the use of the PnP algorithm in a flight environment through two novel uses of existing navigation state information. First, by using the attitude estimate from the INS to reduce the PnP problem from 6 degrees of freedom to 3, the 3D RMS error of the position estimate returned by PnP was reduced roughly by a factor of 15, on average. Second, by using the altitude of the aircraft, along with the camera calibration model, the scale of the observed image can be used to improve the matching performance, generating more inlier 2D/3D correspondences than when naively using all the keypoints from the observed image, generating additional PnP position estimates.

In addition, this chapter evaluated the performance of the PnP position estimation algorithm using multiple keypoint detectors and descriptor extractors. Specifically, the BRISK and SIFT keypoint detectors and descriptor vectors were used in the PnP position estimation process to generate consistently accurate estimates of aircraft position. The BRISK keypoint detector and descriptor vector was evaluated across multiple flights to generate 2237 position estimates, with a mean 3D RSS error of 5.3 m.

Furthermore, this chapter outlined a process to use the artifacts of the PnP position estimation process, 2D/3D correspondences to refine the intrinsic and extrinsic calibration parameters of the camera. This process was also used to estimate of global position offset in the landmark database, which when corrected, significantly improved subsequent employments of the PnP algorithm for pose estimation.

VI. Conclusion

The goal of the research presented in this dissertation is to provide absolute position updates to an airborne platform by comparing information extracted from airborne imagery to a reference database. This was accomplished through the implementation of a novel reference-image derived landmark database, and two position-estimation algorithms.

A method for creation of an image keypoint database created from orthorectified reference imagery was developed. The database was also shown to supports an offline machine-learning process to develop a heuristic, NavRank, which predicts how likely a landmark is to be used in the online navigation algorithms detailed in this dissertation. The effectiveness of NavRank was demonstrated by maintaining equivalent navigation performance with databases 10% the size of a naively created database. In addition, this dissertation showed that the database is able to recover from error in the landmark positions resulting from a global bias in the georegistration process, improving the navigation performance of subsequent flights.

This landmark database was then used in two online navigation algorithms. The first algorithm was a novel wide-area search algorithm, designed to provide a rough estimate of aircraft position for recovery in situations where the the prior position estimate has a large associated uncertainty. This algorithm implemented a histogram filter that uses a novel measurement likelihood function, which was approximated using statistics of matches between keypoints detected in imagery observed by the airborne platform, and the reference database. This algorithm was demonstrated using actual flight data, and shown to be able to recover aircraft position over a large, 45 km $\times$ 55 km search extent.

The resulting coarse pose was then used to initialize a fine-tracking function, which provides GPS-like position estimates. This research showed that careful sampling of keypoints using metadata provided by the database provides a significant matching benefit. Furthermore, it was shown that using the attitude estimate from the INS to constrain the six degrees of freedom DOF position and attitude estimation problem to three-DOF provides significant position accuracy improvement. This research also addressed another key challenge of airborne navigation – camera calibration – and demonstrated that an accurate camera calibration can be accomplished using the airborne platform observations along with the landmark database. Analysis of flight data showed that this algorithm is capable of providing GPS-level position estimates, when used in conjunction with an IMU.

6.1 Future Work

The large scope of the research described in this dissertation opens several avenues of future research in each of the main contribution areas. The framework developed for analysis of the landmark database is well suited for the further development of statistics that measure landmark usefulness for navigation. For example, this research did not address statistics with respect to the seasonal variability of landmark observability, or the variability conditioned on an individual platform or sensor. Additionally, the landmark classification algorithm derived in Chapter III could be used in conjunction with the landmark clustering approaches found in the Content Based Image Retrieval (CBIR) works, to develop more informative visual vocabulary models. While a model for updating the landmark positions and NavRank metrics were discussed, additional work should address rigorous models for adding and removing landmarks from the database, along with updating the appearance model, conditioned on new observations.

As more systems utilize vision-aided navigation in a flight environment, wide-area search for position acquisition in situations where a large uncertainty about the prior pose estimate exists becomes more essential. There is significant room for improvement of the algorithm presented in this dissertation, specifically by investigating how to best utilize the benefits of binary keypoint descriptors like BRISK. With the ability to hold 8 times as many BRISK descriptor vectors in the same amount of physical memory as SIFT descriptor vectors, the search extent defined in Chapter IV could hold significantly more information. With the orders of magnitude reduction in the time it takes to compare descriptor vectors, the approximation of the observation likelihood used in the histogram filter in Chapter IV could also incorporate a geometric constraint to improve matching performance. Furthermore, the results of the algorithm were evaluated only for a single flight campaign, where each flight had relatively similar trajectories.

While the fine-tracking algorithm developed in Chapter V is relatively mature, there is still some room for improvement. After performing automatic camera calibration, and updating the landmark database for an estimated local bias, the results in Chapter V show that there is still a slight (4.5m) average error in the down direction. This research did not investigate the existence of any bias of landmark position estimates in an area local to the current observation. We suggest that augmentation of the PnP state vector to include local position offset may help reduce error in the vertical direction. In addition, while an approach for estimation of the uncertainty of the position estimate from the 3 DoF PnP solution was discussed, further work is needed to resolve the overly optimistic estimates of horizontal uncertainty.

Furthermore, this research was conducted entirely using features of both the reference satellite imagery, and the observed platform imagery. While the use of features helps mitigate problems with storage, and computational complexity when performing inference using imagery, some information is still lost. Information loss can make some tasks intractable, such as deriving rigorous decisions about the integrity of a position solution estimate computed using image features. Initial investigation into the use of probabilistic, volumetric models of the structure and appearance of the world provide an elegant path to mitigation of information loss incurred when using keypoints [51].

Bibliography

1. R. Hartley and A. Zisserman, *Multiple view geometry in computer vision.* Cambridge Univ Press, 2000, vol. 2.
2. D. Titterton and J. L. Weston, *Strapdown inertial navigation technology.* IET, 2004, vol. 17.
3. P. D. Groves, *Principles of GNSS, inertial, and multisensor integrated navigation systems.* Artech house, 2013.
4. M. M. Veth and J. Raquet, "Fusing low-cost image and inertial sensors for passive navigation," *Navigation*, vol. 54, no. 1, pp. 11–20, 2007.
5. W. D. Rencken, "Autonomous sonar navigation in indoor, unknown and unstructured environments," in *Intelligent Robots and Systems' 94.'Advanced Robotic Systems and the Real World', IROS'94. Proceedings of the IEEE/RSJ/GI International Conference on*, vol. 1. IEEE, 1994, pp. 431–438.
6. L. Engelbrecht and A. Weinberg, "Location determination system and method using television broadcast signals," Apr. 23 1996, US Patent 5,510,801.
7. A. Varshavsky, M. Y. Chen, E. de Lara, J. Froehlich, D. Haehnel, J. Hightower, A. LaMarca, F. Potter, T. Sohn, K. Tang *et al.*, "Are gsm phones the solution for localization?" in *Mobile Computing Systems and Applications, 2006. WMCSA'06. Proceedings. 7th IEEE Workshop on.* IEEE, 2005, pp. 34–42.
8. J. Campbell, M. Uijt de Haag, and F. Van Graas, "Terrain-referenced positioning using airborne laser scanner," *Navigation*, vol. 52, no. 4, pp. 189–197, 2005.
9. K. Kauffman, J. Raquet, Y. Morton, and D. Garmatyuk, "Real-time UWB-OFDM radar-based navigation in unknown terrain," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 49, no. 3, pp. 1453–1466, 2013.
10. S. Thrun, W. Burgard, and D. Fox, *Probabilistic robotics.* MIT press, 2005.
11. J. Sivic and A. Zisserman, "Video google: A text retrieval approach to object matching in videos," in *Computer Vision, 2003. Proceedings. Ninth IEEE International Conference on.* IEEE, 2003, pp. 1470–1477.
12. M. Cummins, "Probabilistic localization and mapping in appearance space," Ph.D. dissertation, University of Oxford, 2009.
13. A. R. Zamir and M. Shah, "Accurate image localization based on google maps street view," in *Computer Vision–ECCV 2010.* Springer, 2010, pp. 255–268.
14. P. Turcot and D. G. Lowe, "Better matching with fewer features: The selection of useful features in large database recognition problems," in *Computer Vision Workshops (ICCV Workshops), 2009 IEEE 12th International Conference on.* IEEE, 2009, pp. 2109–2116.

15. "Department of defense world geodetic system 1984, its definition and relationships with local geodetic systems," National Geospatial Agency, National Geospatial-Intelligence Agency (NGA) Standarization Document; NGA.STND.0036, 2014.

16. J. T. Sample and E. Ioup, *Tile-based geospatial information systems: principles and practices.* Springer Science & Business Media, 2010.

17. P. S. Maybeck, *Stochastic models, estimation, and control.* Academic press, 1982, vol. 3.

18. M. J. Veth, R. K. Martin, and M. Pachter, "Anti-temporal-aliasing constraints for image-based feature tracking applications with and without inertial aiding," *IEEE Transactions on Vehicular Technology*, vol. 59, no. 8, pp. 3744–3756, 2010.

19. S. Soatto, "Visual scene representations: Sufficiency, minimality, invariance and approximations," *arXiv preprint arXiv:1411.7676*, 2014.

20. E. Rosten, R. Porter, and T. Drummond, "Faster and better: A machine learning approach to corner detection," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 32, pp. 105–119, 2010. [Online]. Available: http://lanl.arXiv.org/pdf/0810.2434

21. D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International journal of computer vision*, vol. 60, no. 2, pp. 91–110, 2004.

22. N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 1. IEEE, 2005, pp. 886–893.

23. A. Alahi, R. Ortiz, and P. Vandergheynst, "Freak: Fast retina keypoint," in *2012 IEEE Conference on Computer Vision and Pattern Recognition (CVPR).* Ieee, 2012, pp. 510–517.

24. E. D. Kaplan and C. J. Hegarty, *Understanding GPS: principles and applications.* Artech house, 2005.

25. A. I. Mourikis and S. I. Roumeliotis, "A dual-layer estimator architecture for long-term localization," in *2008 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops.* IEEE, 2008, pp. 1–8.

26. M. Kaess, H. Johannsson, R. Roberts, V. Ila, J. Leonard, and F. Dellaert, *Intl. J. of Robotics Research, IJRR.*

27. K. Konolige, J. Bowman, J. Chen, P. Mihelich, M. Calonder, V. Lepetit, and P. Fua, "View-based maps," *The International Journal of Robotics Research*, 2010.

28. J. McDonald, M. Kaess, C. Cadena, J. Neira, and J. J. Leonard, "Real-time 6-dof multi-session visual slam over large-scale environments," *Robotics and Autonomous Systems*, vol. 61, no. 10, pp. 1144–1158, 2013.

29. H. Bay, T. Tuytelaars, and L. Van Gool, "Surf: Speeded up robust features," in *Computer Vision–ECCV 2006*. Springer, 2006, pp. 404–417.

30. S. Leutenegger, M. Chli, and R. Y. Siegwart, "Brisk: Binary robust invariant scalable keypoints," in *2011 IEEE International Conference on Computer Vision (ICCV)*. IEEE, 2011, pp. 2548–2555.

31. E. Rosten and T. Drummond, "Machine learning for high-speed corner detection," in *Computer Vision–ECCV 2006*. Springer, 2006, pp. 430–443.

32. T. Lindeberg, "Image matching using generalized scale-space interest points," *Journal of Mathematical Imaging and Vision*, vol. 52, no. 1, pp. 3–36, 2015.

33. F. Alted, I. Vilata, S. Prater, V. Mas, T. Hedley *et al.*, "Pytables: hierarchical datasets in python," *Available from World Wide Web: http://www. pytables. org*, 2002.

34. M. A. Aguilar, M. del Mar Saldaña, and F. J. Aguilar, "Assessing geometric accuracy of the orthorectification process from geoeye-1 and worldview-2 panchromatic images," *International Journal of Applied Earth Observation and Geoinformation*, vol. 21, pp. 427–435, 2013.

35. "Spatial referencing by coordinates," International Organization for Standardization, Geographic information ISO 19111:2007, 2007.

36. USGS, *1 Arc Second Scene, Shuttle Radar Topography Mission.* Global Land Cover Facility, University of Maryland, 2004.

37. F. G. Lemoine, S. C. Kenyon, J. K. Factor, R. G. Trimmer, N. K. Pavlis, D. S. Chinn, C. M. Cox, S. M. Klosko, S. B. Luthcke, M. H. Torrence *et al.*, "The development of the joint NASA GSFC and the national imagery and mapping agency (NIMA) geopotential model EGM96," 1998.

38. F. Alted, "Why modern cpus are starving and what can be done about it," *Computing in Science & Engineering*, vol. 12, no. 2, pp. 68–71, 2010.

39. M. Muja and D. G. Lowe, "Fast approximate nearest neighbors with automatic algorithm configuration." in *VISAPP (1)*, 2009, pp. 331–340.

40. J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman, "Object retrieval with large vocabularies and fast spatial matching," in *2007 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2007, pp. 1–8.

41. E. K. Donald, "The art of computer programming," *Sorting and searching*, vol. 3, pp. 426–458, 1999.

42. G. Bradski *et al.*, "The opencv library," 2000.

43. S. Van Der Walt, J. L. Schönberger, J. Nunez-Iglesias, F. Boulogne, J. D. Warner, N. Yager, E. Gouillart, and T. Yu, "scikit-image: image processing in python," *PeerJ*, vol. 2, p. e453, 2014.

44. M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Y. Ng, “Ros: an open-source robot operating system,” in *ICRA workshop on open source software*, vol. 3, no. 3.2, 2009, p. 5.

45. C. Wu, “SiftGPU: A GPU implementation of scale invariant feature transform (sift)(2007),” *URL http://cs. unc. edu/˜ ccwu/siftgpu*, 2011.

46. B. Triggs, P. F. McLauchlan, R. I. Hartley, and A. W. Fitzgibbon, “Bundle adjustment—a modern synthesis,” in *Vision algorithms: theory and practice.* Springer, 2000, pp. 298–372.

47. V. Lepetit, F. Moreno-Noguer, and P. Fua, “Epnp: An accurate O(n) solution to the PnP problem,” *International journal of computer vision*, vol. 81, no. 2, pp. 155–166, 2009.

48. K. Levenberg, “A method for the solution of certain problems in least squares,” *Quarterly of applied mathematics*, vol. 2, pp. 164–168, 1944.

49. S. Agarwal, Y. Furukawa, N. Snavely, I. Simon, B. Curless, S. M. Seitz, and R. Szeliski, “Building Rome in a day,” *Communications of the ACM*, vol. 54, no. 10, pp. 105–112, 2011.

50. M. A. Fischler and R. C. Bolles, “Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography,” *Communications of the ACM*, vol. 24, no. 6, pp. 381–395, 1981.

51. D. E. Crispell, “A continuous probabilistic scene model for aerial imagery,” Ph.D. dissertation, Brown University, 2010.

REPORT DOCUMENTATION PAGE	*Form Approved OMB No. 0704-0188*

1. REPORT DATE *(DD-MM-YYYY)* 15-09-2016	**2. REPORT TYPE** Doctoral Dissertation	**3. DATES COVERED** *(From - To)* 01-09-2013 - 15-09-2016

4. TITLE AND SUBTITLE Improving Real-World Performance of Vision Aided Navigation in a Flight Environment	**5a. CONTRACT NUMBER**
	5b. GRANT NUMBER
	5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S) Venable, Donald T (AFRL/RYWN)	**5d. PROJECT NUMBER** N/A
	5e. TASK NUMBER
	5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Air Force Institute of Technology Graduate School of Engineering and Management (AFIT/EN) 2950 Hobson Way, Wright-Patterson AFB OH 45433-7765	**8. PERFORMING ORGANIZATION REPORT NUMBER** AFIT-ENG-DS-16-S-017
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) Jeffrey Hebert; jeffrey.hebert.1@us.af.mil; (937) 713-4418 Navigation and Communications Branch, Sensors Directorate Air Force Research Laboratory 2241 Avionics Cir, WPAFB, OH 45433	**10. SPONSOR/MONITOR'S ACRONYM(S)** AFRL/RYWN
	11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT

Distribution Statement A. Approved for Public Release;Distribution Unlimited

13. SUPPLEMENTARY NOTES

This work is declared a work of the U.S. Government and is not subject to copyright protection in the United States.

14. ABSTRACT

The motivation of this research is to fuse information from an airborne imaging sensor with information extracted from satellite imagery in order to provide accurate position when GPS is unavailable for an extended duration. A corpus of existing georeferenced satellite imagery is used to create a keypoint database. A novel algorithm for recovering coarse pose using by comparing keypoints extracted from the airborne imagery to the reference database is developed. This coarse position is used to bootstrap a local-area georegistration algorithm, which provides GPS-level position estimates.

15. SUBJECT TERMS

Navigation, Image Processing, Computer Vision, Image Georegistration

16. SECURITY CLASSIFICATION OF:			**17. LIMITATION OF ABSTRACT**	**18. NUMBER OF PAGES**	**19a. NAME OF RESPONSIBLE PERSON**
a. REPORT	**b. ABSTRACT**	**c. THIS PAGE**			Dr. John F. Raquet, AFIT/ENG
U	U	U	150	150	**19b. TELEPHONE NUMBER** *(Include area code)* 937-255-3636 x4580; john.raquet@afit.edu

Standard Form 298 (Rev. 8/98)
Prescribed by ANSI Std. Z39.18